



UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 1

**Sistema de gestión y aseguramiento material para la  
Organización Nacional de Bufetes Colectivos**

Trabajo de diploma para optar por el título de  
Ingeniero en Ciencias Informáticas

**Autor:** Maykol Daniel González Matos

**Tutores:**

**MSc. PA** Leiny Amel Pons Flores

**Ing. PA** Serguey Gonzalez Garay

**La Habana, diciembre de 2021**

**“Año 63 de la Revolución**



*Albert Einstein*

## DECLARACIÓN DE AUTORÍA

---

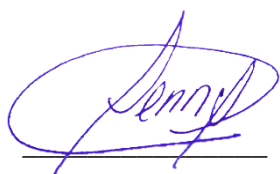
Declaro por este medio que yo, Maykol Daniel González Matos, con carné de identidad 97112106622, soy el autor principal del trabajo titulado “Sistema de gestión y aseguramiento material para la Organización Nacional de Bufetes Colectivos” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos el presente documento a los \_\_ días del mes de \_\_\_\_\_ del año 2021.



**Firma del Autor**

Maykol Daniel González Matos



**Firma del Tutor**

**MsC. PA** Leiny Amel Pons Flores



**Firma del Tutor**

**Ing. PA** Serguey Gonzalez Garay

## DEDICATORIA

---

*A todos los que de una forma u otra estuvieron en este grandioso camino, por confiar siempre en mí, motivarme a convertirme en una mejor persona, por apoyar cada uno de mis pasos, por su entrega y dedicación y por ser sencillamente especiales.*

## AGRADECIMIENTO

---

*En especial a MI MADRE, gracias te doy por todos tus consejos y el apoyo, gracias por confiar en mí y decirme innumerables veces que yo podía, gracias por no dejarme solo nunca, gracias por sufrir por mí y llorar tantas veces por no verme llorar a mí, gracias por darme la vida y dejarme ser quien soy sin imponerme nada. Hoy llego al fin de uno de mis caminos y te dedico este logro a ti por todo el sacrificio que has realizado, y espero que te sientas orgullosa de tu hijo que ¡Te Ama!*

*¡¡¡A Mi Papá CHECHÉ!!!, hombre inigualable en mi vida, consejero en cada uno de los momentos buenos y malos. Sé que eres un hombre fuerte y duro, pero con el corazón más grande que he conocido. Tú has sido mi primer ejemplo y serás al único que deseo aspirar. Te dedico de forma muy personal este gran logro que deseabas desde hacía mucho. No has tenido descanso conmigo, y hoy quiero que te sientas más tranquilo, ya soy ingeniero y fue en gran medida gracias a ti. ¡Te amo!*

*A mi abuela Gladis, agradecido estoy por todo lo que me has enseñado cuando era niño y por todo el esfuerzo que hiciste y todavía haces por tus nietos. Con tu cosas y achaques de la vida quiero que sepas que tu nieto te ama mucho y siempre estaré orgulloso de decirte Abue. ¡Te amo!*

A mi abuela Caridad, mi flaca preferida, gracias por tus cuidados de niño y todo el tiempo que me dedicaste, te debo mucho en la vida por todo el tiempo que me dedicaste a mi hermana y a mí, eso nunca lo vas a recuperar y no tengo forma de pagártelo, más que diciéndote que ¡Te amo!

A mi hermana Daily, por ser mi compañera de vida y siempre confiar en mí, gracias por darme ese niño tan lindo, que, aunque sea tu hijo desde que nació también paso a ser parte de mi vida. Muchas veces estamos molestos y discutimos, pero quiero que entiendas que si soy así es por el amor que te tengo. Por ti daría mi vida y más, y sufro cuando tu sufres. Los pasos que doy en mi vida por muy grande que sean para ti, piensa siempre que yo estoy dándolos porque tu estas en mi mente y nunca te voy a dejar de lado. ¡Te amo!

A mi sobrino, todavía eres muy pequeño para entender las palabras que puedo decirte, pero algún día serás grande y las podrás leer. No dormí el día que nacientes y no lo hare jamás. Le has dado un sentido diferente a mi vida, y de todos los cambios tú has el mejor. Gracias por llegar y ser tan especial. ¡Te amo!

A mi Novia Milena, gracias por todo el apoyo especial y esfuerzo que has hecho para que hoy alcance un triunfo. Gracias por las noches en que no te deje casi dormir por tener q estudiar, por todos los días malos de tesis que tuvimos juntos. No ha sido nada fácil

*alcanzar la meta, pero sin ti me hubiera sido más difícil. Te agradezco cada uno de los detalles y ayudas que me has dado, hoy nuestras vidas cambian completamente y cogerán un rumbo lleno de éxitos personales que disfrutaremos juntos. Gracias por la paciencia y la dedicación que me has tenido. ¡Te amo!*

*A los profesores que, desde el inicio de la carrera, con sus consejos, su enseñanza y dedicación, perfeccionaron aquel estudiante para convertirse hoy en un futuro ingeniero, muchas gracias por su apoyo en todo momento y por su amistad más que todo. En especial a todos los que me hicieron sentir en familia, las noches de comidas, los momentos, los regaños y los apoyos.*

*¡¡¡ Quiero agradecer de forma especial a todas las personas que me apoyaron y me dieron la fuerza que necesité para lograr que este sueño se hiciera realidad!!!*

## RESUMEN.

---

La presente investigación está encaminada a solucionar los problemas referentes al proceso de gestión y aseguramiento material en la Organización Nacional de Bufetes Colectivos; a partir de un sistema web que permita la consolidación de la economía en la entidad. El estudio y análisis del estado del arte permitieron identificar funcionalidades y tecnologías que facilitaron la comprensión y diseño de la solución propuesta. El desarrollo estuvo guiado por la metodología AUP-UCI y se seleccionó como principales tecnologías el marco de trabajo Laravel, PHP como lenguaje de programación, el entorno integrado de desarrollo *Visual Studio Code* y como gestor de base de datos se utilizó PostgreSQL. La estrategia de prueba planteada, permitirá verificar el cumplimiento de los objetivos trazados y evaluar la calidad del sistema.

**Palabras clave:** aseguramiento, gestión, problemas de investigación, sistema de gestión, sistema web.



## ÍNDICE

---

INTRODUCCIÓN.....	1
CAPÍTULO 1: “FUNDAMENTACIÓN TEÓRICA” .....	7
1.1 Sistema de gestión de almacén (SGA).....	7
1.2 Estudio de sistemas homólogos .....	10
1.2.1 Estudio de sistemas homólogos existentes en el ámbito nacional .....	11
VERSAT.....	11
RODAS XXI .....	11
Suite Zun.....	12
SUITE FACSI+.....	13
Distra .....	13
1.2.2 Estudio de sistemas homólogos existentes en el ámbito internacional.....	14
EXACT.....	14
Easy WMS .....	14
AHORA .....	15
Conclusión de los Sistemas de Gestión de Almacén (SGA) .....	16
1.3 Metodología de desarrollo de software .....	16
Metodología de desarrollo de software AUP versión UCI.....	17
1.4 Tecnologías y herramientas para el desarrollo. ....	19
1.4.3. Lenguajes de Programación .....	19
Lenguaje del lado del cliente.....	19
Lenguaje <i>HyperText Markup Language</i> (HTML) 5.....	19
Lenguaje Cascading Style Sheets 3 (CSS) .....	20
Bootstrap 4.....	20
JavaScript (JS).....	20
Angular V 13.0.2 .....	21
Lenguaje de lado del servidor .....	21
Laravel V 8.0.....	22
Sistema gestor de base de datos.....	22
PostgreSQL V 9.4 .....	22
Servidor de aplicaciones web .....	22
Apache V 2.4.46.....	22
Entorno de desarrollo .....	23
Visual Studio Code V 1.62.3.....	23
Lenguaje de modelado.....	23

Lenguaje Unificado de Modelado (UML) V 2.0.....	23
Herramienta de modelado .....	23
Herramientas de validación .....	23
1.5 Conclusiones del capítulo.....	24
<b>CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA DE GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”</b> .....	<b>24</b>
2.1 Introducción.....	24
2.2 Descripción de la propuesta de solución .....	24
2.3 Modelo Conceptual .....	25
2.4 Diagrama de caso de uso del sistema.....	26
2.5 Diagramas de clases del diseño .....	27
2.6 Diagrama de secuencia .....	28
2.7 Modelo de datos.....	29
2.8 Requisitos de la propuesta de solución .....	30
2.8.1 Requisitos funcionales .....	30
2.8.2 Requisitos No Funcionales .....	32
2.9 Arquitectura de software.....	33
2.10 Patrones de Diseño .....	35
2.11 Conclusiones del capítulo.....	36
<b>CAPÍTULO 3: “IMPLEMENTACIÓN Y VALIDACIÓN DEL SISTEMA PARA LA DE GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”</b> .....	<b>35</b>
3.1 Diagrama de despliegue.....	35
Descripción de elementos e interfaces de comunicación .....	36
3.2. Estándares de codificación.....	36
Etiquetas de apertura y cierre .....	37
Identación .....	37
Operadores.....	38
Uso de comillas .....	38
Uso de punto y coma (;) en código PHP .....	39
Estructuras de control .....	39
Funciones .....	41
Arreglos ( <i>Arrays</i> ).....	42
Nombres de archivos .....	42
Comentar el código.....	43
3.3 Estrategia de Pruebas .....	43
Pruebas unitarias.....	45

Pruebas funcionales.....	45
Pruebas de aceptación .....	48
Pruebas de integración .....	49
Pruebas de sistema .....	49
Pruebas no funcionales.....	49
Pruebas de rendimiento (carga y estrés) .....	49
Pruebas de rendimiento.....	50
Pruebas de seguridad.....	51
Pruebas de usabilidad .....	51
3.4 Conclusiones del capítulo.....	52
CONCLUSIONES .....	50
REFERENCIA BIBLIOGRÁFICA.....	51
ANEXOS.....	55

## ÍNDICE DE TABLAS

---

Tabla 1 Resultado del estudio de Homólogos [Elaboración Propia] .....	15
Tabla 2 Fases de la variación de AUP para la UCI [Elaboración propia].....	17
Tabla 3 Descripción de requisitos funcionales [Elaboración propia].....	30
Tabla 4 Descripción de requisitos no funcionales [Elaboración propia].....	32
Tabla 5 Casos de prueba: Añadir usuario [Elaboración propia]; <b>Error! Marcador no definido.</b>	
Tabla 6 Caso de pruebas Editar usuario [Elaboración propia].....	56

## ÍNDICE DE FIGURAS

---

Figura 1 Sistema Nacional de Planificación de Cuba. Fuente: Comisión Económica para América Latina y el Caribe (CEPAL).....	2
Figura 2 Escenarios de la metodología AUP-UCI [Elaboración propia].....	18
Figura 3 Descripción de la propuesta de solución [Elaboración propia] .....	25
Figura 4 Arquitectura MVC [Elaboración propia] .....	35
Figura 5 Ejemplo de etiquetas de apertura y cierre (PHP: Hypertext Preprocessor, 2021) .....	37
Figura 6 Ejemplo de indentación (PHP: Hypertext Preprocessor, 2021) .....	38
Figura 7 Ejemplo de operadores (PHP: Hypertext Preprocessor, 2021). .....	38
Figura 8 Ejemplo de comillas (PHP: Hypertext Preprocessor, 2021).....	39
Figura 9 Ejemplo del uso del (;) (PHP: Hypertext Preprocessor,2021).....	39
Figura 10 Ejemplo de while (PHP: Hypertext Preprocessor, 2021). .....	40
Figura 11 Ejemplo de if (PHP: Hypertext Preprocessor, 2021).....	40
Figura 12 Ejemplo de casos en que no es obligatorio el uso de ( {} ) (PHP: Hypertext Preprocessor, 2021) .....	41
Figura 13 Ejemplo de elseif ( PHP: Hypertext Preprocessor, 2021). .....	41
Figura 14 Ejemplo de funciones ( PHP: Hypertext Preprocessor, 2021). .....	42
Figura 15 Ejemplo de array (PHP: Hypertext Preprocessor,2021). .....	42
Figura 16 Ejemplo de comentarios en el código (PHP: Hypertext Preprocessor, 2021) .....	43
Figura 1 Resultados de las iteraciones de las pruebas funcionales al gestionar usuario.....	47

## ÍNDICE DE DIAGRAMA

---

Diagrama 1 Modelo Conceptual [Elaboración propia] .....	26
Diagrama 2 Diagrama de casos de uso del sistema [Elaboración propia] .....	27
Diagrama 3 DCD con estereotipos web Gestionar Insumos [Elaboración propia] .....	28
Diagrama 4 Diagrama de secuencia Crear Insumo [Elaboración propia] .....	29
Diagrama 5 Modelo de datos [Elaboración propia].....	30
Diagrama 6 Diagrama de despliegue [Elaboración propia] .....	35

### INTRODUCCIÓN

Los sistemas económicos son aquellos métodos de producción asumidos por los países y gobiernos que conllevan un conjunto de acciones que contribuyen al desarrollo económico de las sociedades.

Se pueden clasificar de acuerdo a varios criterios, pero, de forma general, los tipos de sistemas económicos actuales son el de economía planificada, el de economía capitalista y el de economía mixta.

Carlos Marx vio a la economía planificada como aquella en que el mercado tiene un papel secundario para la asignación de recursos, y que estos deben estar en articulación directa con las necesidades presentes y sobre todo futuras de la economía de un país (Arboleda Aparicio, 2018).

En una economía centralmente planificada, el modelo económico o Sistema de Dirección de la Economía (SDE) se define a partir de los 3 subsistemas que lo componen: a) la planificación; b) la gestión; c) los estímulos. Con la planificación se determinan las relaciones entre el órgano planificador, los organismos centrales de la administración estatal y las unidades productivas. Inicialmente, la planificación es un proceso de definición de objetivos económicos y de los recursos para conseguirlos (en términos físicos o monetarios) a nivel de gobierno, luego desagregados hacia los ministerios, los organismos y las empresas (Bruno Sovilla y Francisco García Fernández, 2013).

La planificación de la economía está muy relacionada con el presupuesto del estado. El control del presupuesto se asocia de manera inseparable a otros procesos como la planificación, organización y ejecución; solo con un enfoque integrador es posible la detección y corrección de las desviaciones con una visión preventiva e integradora (Universidad Pedagógica Enrique José et al., 2017).

La actividad logística está estrechamente relacionada con la planificación de la economía. Esta actividad se ha convertido en un proceso vital para lograr que las empresas se posicionen en el mercado competitivo de estos tiempos y escalen a posiciones dominantes en aspectos como rentabilidad y competitividad.

El perfeccionamiento de la gestión del sistema logístico contribuye a que las organizaciones incrementen la producción, las ventas, la calidad de los servicios, y aumenten las utilidades, lo que permite, incrementar el capital, para realizar

inversiones en los negocios; garantizando un mayor número de empleo, y un mejoramiento de las condiciones de vida y de trabajo de los integrantes de la organización, y la satisfacción de los clientes (Ruano-Ortega, 2020).

Cuba trabaja sobre la base de una economía planificada. Inicialmente tardó más de 15 años y tuvo que realizar múltiples experimentos y revisiones “sobre la marcha” de premisas básicas hasta poder ejecutar, al fin, su primer plan quinquenal para el período 1976–1980, con técnicas de planificación al estilo soviético. Aquello que el economista húngaro János Kornai definió como la “forma madura” del sistema socialista clásico, se convirtió, finalmente, en una realidad en Cuba (Budrich GmbH, 2021).

En Cuba, la planificación centralizada es la categoría definitoria del sistema de dirección, a través de la cual el Estado tiene la posibilidad, en representación de los intereses de toda la sociedad, de conducir conscientemente el proceso de desarrollo económico y social del país e inducir las acciones de todos los actores económicos en función de los objetivos definidos (CEPAL, 2021).

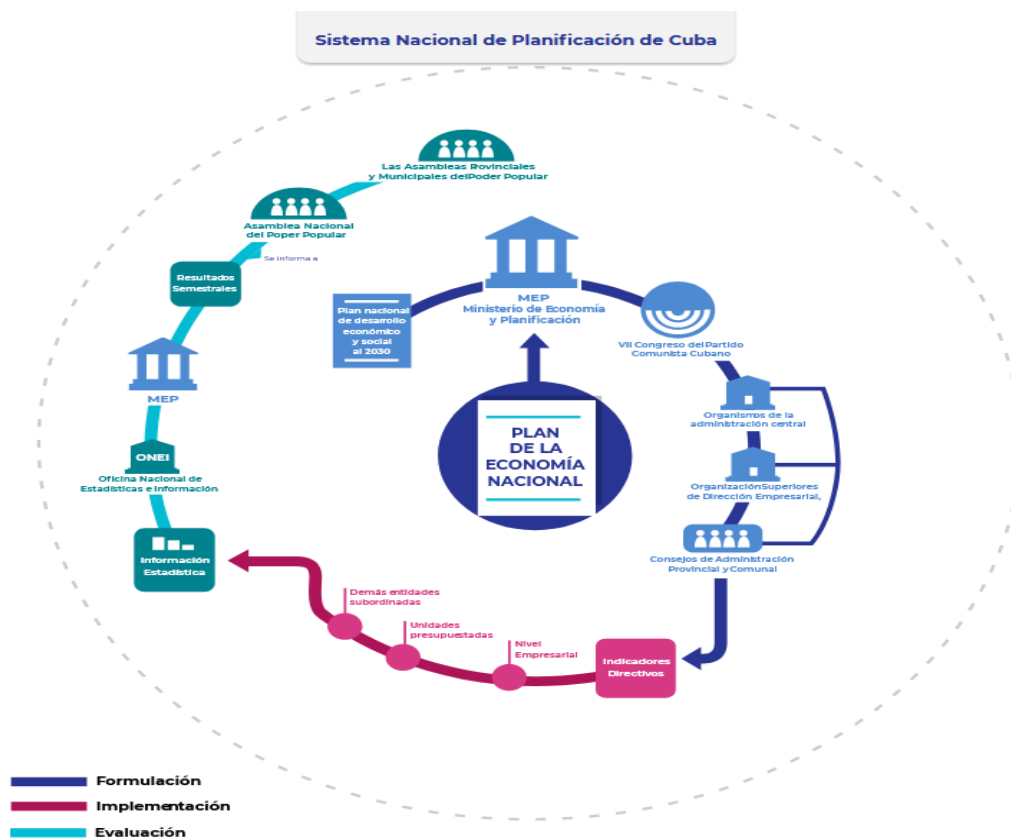


Figura 2 Sistema Nacional de Planificación de Cuba. Fuente: Comisión Económica para América Latina y el Caribe (CEPAL).



La conducción de la elaboración de los planes anualmente se sustenta en las “Las Indicaciones Metodológicas para la elaboración del Plan de la Economía Nacional”. Dichas indicaciones son objeto de revisión anualmente, en atención a las transformaciones que se suceden en el sistema de dirección en correspondencia con la actualización del modelo economía social y los Lineamientos de la Política Económica y Social del Partido y la Revolución, aprobados en el VII Congreso del Partido Comunista de Cuba.

En las Indicaciones Metodológicas en cuestión se inscribe la participación en la elaboración del Plan de la Economía Nacional de los organismos de la Administración Central del Estado, las Organizaciones Superiores de Dirección Empresarial y los Consejos de la Administración Provincial, así como otras entidades económicas, puntualizando la información y base cálculo a brindar por cada uno de ellos, entre otras cuestiones, el flujo de tareas y coordinaciones que se generan, así como el cronograma de trabajo para su cumplimiento (CEPAL, 2021).

Es habitual en Cuba que cada año se realice el proceso de planificación de la economía, rectorado por el Ministerio de Economía y Planificación y desarrollado por todos los organismos y entidades del estado y el gobierno. Estos organismos y entidades dirigen metodológicamente a otras instituciones como la Organización Nacional de Bufetes Colectivos (ONBC).

Como parte del proceso de planificación de la economía que todos los años realizan las empresas, entidades e instituciones del país, en la ONBC se despliega una serie de actividades organizativas para realizar esta importante tarea. Hace varios años se trabaja con un libro Excel para modelar todo lo referente a las necesidades materiales imprescindibles para el correcto funcionamiento de la ONBC. Este archivo se distribuye de forma individual para cada provincia, donde luego de ser llenados los datos necesarios se envía a la Junta Directiva para realizar el consolidado del mismo a nivel nacional.

La ONBC es una entidad autopresupuestada del sector estatal que posee más de 187 bufetes en todo el país y 16 unidades de administración provincial. Esta Organización administra un almacén nacional al que se le realizan solicitudes por los directivos autorizados para extraer materiales de todo tipo, y se llena un modelo (comprado en imprenta) de 3 copias por cada material a retirar.

Inmerso el país y su economía en profundos cambios de la Tarea Ordenamiento, se ha producido el alza de los precios de todos los materiales e insumos que se utilizan en la ONBC. Entre los productos más consumidos se encuentra el papel y los bolígrafos, los cuales son utilizados en grandes cantidades para llenar los vales de solicitud de salida del almacén, debido a que este proceso no se encuentra informatizado.

Durante años se han presentado dificultades con el proceso de consolidado debido a que en las provincias se modifica el archivo original, agregando o eliminando contenido. Esto provoca que los libros no puedan ser consolidados de forma automática, involucrando personal para realizar el trabajo de forma manual, sujeto a errores y consumiendo un mayor tiempo en la actividad. También se modifican las fórmulas del libro dejando la posibilidad a introducir errores de cálculos que afecten los datos económicos y que deban revisarse una vez concluidos de forma manual.

Todos estos elementos sobre las dificultades que presenta la ONBC en la gestión y aseguramiento material permiten definir como **problema de investigación**: ¿Cómo contribuir al proceso de gestión y aseguramiento material de la Organización Nacional de Bufetes Colectivos?

Se define como **objetivo de estudio**: La gestión y aseguramiento material institucional.

Se plantea como **objetivo general**: Desarrollar un sistema de gestión y aseguramiento material para contribuir al proceso de consolidación en la Organización Nacional de Bufetes Colectivos, utilizando tecnologías web.

Se establece como **campo de acción**: La gestión y aseguramiento material en la Organización Nacional de Bufetes Colectivos.

### **Objetivos específicos:**

1. Caracterizar los fundamentos teóricos relacionados con las herramientas y sistemas de gestión y aseguramiento material.
2. Definir las tecnologías, las herramientas y la metodología para la implementación del sistema que permita contribuir a la gestión y aseguramiento material en la Organización Nacional de Bufetes Colectivos.

3. Implementar el sistema que permita contribuir a la gestión y aseguramiento material en la Organización Nacional de Bufetes Colectivos.
4. Validar mediante las pruebas definidas, así como los instrumentos científicos necesarios, el correcto funcionamiento y pertinencia del sistema desarrollado.

Se plantea como **hipótesis de la investigación**:

El sistema web de gestión y aseguramiento material contribuirá al proceso de consolidación en la Organización Nacional de Bufetes Colectivos

Entre los **métodos científicos** utilizados para darle cumplimiento a estas tareas se encuentran:

**Teóricos:**

- I. **Analítico-Sintético:** Permitió conocer y analizar los referentes teóricos sobre los sistemas de gestión y aseguramiento material, analizar los elementos conceptuales asociados al problema de investigación y extraer los más relevantes para la implementación del sistema a desarrollar.
- II. **Histórico-Lógico:** Este método ayudo al entendimiento sobre el avance que ha tenido los sistemas de gestión y aseguramiento material en el mundo, y como se aplica el mismo sobre las TIC.

**Métodos Empíricos:**

- I. **Entrevista:** Se empleó en los encuentros con personal de las distintas áreas de la ONBC para definir las dificultades existentes con respecto a la gestión y aseguramiento material, para así, determinar correctamente los requisitos funcionales y no funcionales del sistema a desarrollar.
- II. **Observación:** Se utilizó con la finalidad de caracterizar y analizar detalladamente cómo se realiza actualmente el proceso de gestión y aseguramiento en la ONBC, identificando los mecanismos para desarrollar un sistema acorde con las dificultades existentes.

La presente investigación se estructurada en 3 capítulos, además de las secciones de Conclusiones, Recomendaciones, Bibliografía y Anexos.

**Capítulo 1: “Estudio del arte y las tecnologías para su desarrollo”:** se realiza un análisis de los principales conceptos y elementos teóricos asociados a la gestión y

aseguramiento material, imprescindibles para la comprensión de la investigación. Se describen las herramientas, tecnologías, conjunto de técnicas y metodología para el desarrollo del sistema.

**Capítulo 2. “Diseño del sistema para la optimización de la gestión y aseguramiento material”:** en este capítulo se expone la propuesta de solución y características del sistema. También se definen los diversos artefactos que especifica el proceso de desarrollo de software utilizado.

**Capítulo 3: “Implementación y prueba del sistema para la optimización de la gestión y aseguramiento material”:** se ejecuta la validación de la solución al problema planteado. Además, se realizan y describen las pruebas de software a la aplicación para garantizar la calidad requerida y que cumpla con los requisitos funcionales establecidos.

## **CAPÍTULO 1: “FUNDAMENTACIÓN TEÓRICA”**

### **1.1 Sistema de gestión de almacén (SGA)**

Actualmente y cada vez más a menudo, la atención al cliente y la gestión de pedidos se realizan de forma online. Este hecho, que resulta de lo más cómodo para el consumidor, supone un reto en lo referente a la logística, todos los actores que intervienen en el proceso de distribución han de ser extremadamente flexibles y contentar a tiempo a sus clientes. Para ello, lo ideal es el uso de un sistema de gestión de almacén.

#### **Sistema de gestión de almacén: funcionalidad y rendimiento**

Para que todos los pedidos se procesen correctamente, se recomienda el uso de un SGA. Este tipo de software o sistema debe contar con dos características esenciales para que lleve a cabo su función de la forma más adecuada posible:

- Que sea escalable respecto al tamaño, rendimiento y funcionalidad.
- Que se comunique con sistemas habituales ERP y con sistemas complementarios de gestión aduanera o de transportes a través de interfaces estándar.

El sistema de gestión de almacén no sólo gestiona el mismo y su stock, sino que también controla los procesos intralogísticos. Porque organiza, prioriza y procesa los pedidos que se reciben a través de los diferentes canales de distribución.

Además, coordina la recepción de mercancías, la entrada en el almacén, la ubicación dentro del mismo, cuándo sale, la preparación del pedido, el embalaje y la expedición. No importa el tamaño del centro logístico, el sistema de gestión de almacén es una herramienta muy útil.

Un sistema de gestión de almacén es complejo y aporta diferentes funciones adicionales al servicio que ya da. No obstante, este mismo sistema, a pesar de controlar las funciones complejas de una empresa, tiene que ser fácil e intuitivo de manejar, incluso para poder controlarlo desde un dispositivo móvil o tablet. Esto hace imprescindible que conste de una interfaz de usuario gráfica que sea fácil de comprender por todos los operarios para su sencillo manejo (LogísticaMC, 2021).

## ¿Qué es la logística?

La logística es el conjunto de actividades que gestiona los flujos materiales, informativos y financieros encaminados a llegar al cliente final, añadiéndole valor al mismo. Garantiza el funcionamiento de los procesos de forma más económica y eficiente posible, partiendo del objetivo de suministrar al cliente los productos y servicios en la cantidad, calidad, plazo, costos y lugar requeridos (Alemán de la Torre et al., 2019).

El impacto de factores tales como la globalización de los mercados, el incremento acelerado de los avances científicos-técnicos, la aceptación acelerada del “justo a tiempo” y el surgimiento de nuevas necesidades como brindar servicios que agreguen valor al producto, permiten operar con costos razonables, por lo que la gestión de los almacenes es una herramienta para mejorar la coordinación demanda – suministro. La gestión de almacenes adquiere gran importancia dentro de la red logística ya que constituyen decisiones claves que definen en gran medida la estructura de los costos - servicios del sistema logístico de una empresa.

En las empresas de hoy, la logística está adquiriendo una importancia cada vez más concientizada desde el punto de vista estratégico, convirtiéndose en un factor determinante para su mejora competitiva y su rentabilidad en un mercado de continuos cambio (León, 2018).

Los sistemas logísticos se organizan para gestionar los flujos primarios y secundarios. Un sistema logístico es una estructura económica compleja formada por elementos y eslabones interconectados en un proceso único de gestión de flujos de materiales, servicios y otros relacionados

Un eslabón del sistema logístico es una determinada entidad económica y/o funcionalmente separada (una subdivisión de la empresa o una empresa jurídicamente independiente) que cumple su objetivo local relacionado con la realización de una o varias actividades logísticas. Los eslabones de un sistema logístico único están unidos por una gestión única del proceso logístico.

Los eslabones de un sistema logístico son los proveedores, los productores, los consumidores y los intermediarios logísticos. Las empresas industriales o comerciales se suelen considerar como una empresa central, los proveedores como primera parte,

mientras que los consumidores como segunda, y los intermediarios logísticos como tercera parte (Ceupe, 2021).

### **Flujo logístico**

El flujo logístico engloba todas las actividades por las que pasa un producto desde la etapa de fabricación hasta la de comercialización, incluyendo el transporte, el almacenaje y la distribución.

Gestionar correctamente estas actividades, que se caracterizan por un constante movimiento de los recursos y de la información, es el primer paso para disponer de una cadena de suministro optimizada.

Lograr un flujo logístico eficiente y que aumente el rendimiento de la cadena de suministro no depende únicamente de las labores propias de la logística (abastecimiento, almacenamiento, preparación de pedidos y distribución), sino también del trabajo coordinado de cada uno de los departamentos de la empresa.

Diseñar un flujo logístico óptimo empieza con un análisis y planificación de la demanda. Sin él, es imposible programar el ritmo de producción, definir la estrategia de transporte hasta el punto de venta, hacer un inventario, etc. (Mecalux, s. f.).

Los inventarios en las empresas juegan un rol importante, a través de su uso, se lleva a cabo el control y ritmo de una producción y el límite tanto del almacenaje de los productos ya elaborados como de las materias primas. Por esta misma razón se puede definir como inventario:

- Al registro documental de los bienes tangibles que se tienen para la venta en el curso ordinario del negocio o para ser consumidos en la producción de bienes o servicios para su posterior comercialización (materias primas, productos en proceso y productos terminados). También se define como un amortiguador entre dos procesos: el abastecimiento y la demanda, donde el proceso de abastecimiento contribuye con bienes al inventario, mientras que la demanda consume el mismo inventario.

El control consiste en el ejercicio del control de las existencias; tanto reales como en proceso de producción y su comparación con las necesidades presentes y futuras, para poder establecer, teniendo en cuenta el ritmo de consumo, los niveles de existencias y las adquisiciones precisas para atender la demanda.

Tener un adecuado registro de inventarios no es simplemente hacerlo porque las empresas grandes lo hacen, porque el contador lo pide o porque los necesitamos para armar un balance general. El objetivo principal es contar con información suficiente y útil para: minimizar costos de producción, aumentar la liquidez, mantener un nivel de inventario óptimo y comenzar a utilizar la tecnología con la consecuente disminución de gastos operativos, así como también conocer al final del período contable un estado confiable de la situación económica de la empresa (Laveriano, 2010).

### **Administración Financiera**

La administración financiera es la disciplina que se ocupa del manejo de los recursos financieros de una empresa, prestando atención sobre su rentabilidad y liquidez.

La administración financiera, de hecho, posee un radio de acción muy amplio dentro de la empresa. Así, tiene a su cargo, movimientos claves de la empresa. Ella tiene a cargo la organización, planificación, dirección y control de los recursos financieros pertenecientes a la empresa.

De manera que toca a esta, toda decisión de inversión, financiamiento, así como las decisiones sobre repartos de dividendos. Por tanto, el papel que viene a desempeñar la administración financiera dentro de la empresa resulta ser fundamental. De ella depende la perdurabilidad en el tiempo de cualquier organización empresarial (Fortún, 2020).

#### **1.2 Estudio de sistemas homólogos**

Registrar de manera sistematizada los hechos contables requiere del desarrollo de herramientas empresariales que permitan y mantengan los registros y controles sistemáticos de todas las operaciones que se realizan en una entidad. Las ventajas del uso de la contabilidad sistematizada sobre la contabilidad manual se evidencian en la rapidez para producir los informes operacionales y de desempeño que requiere la gerencia, la mayor calidad de presentación y seguridad en beneficio de los clientes, proveedores e inversionistas (Pérez Sánchez, 2018)



### **1.2.1 Estudio de sistemas homólogos existentes en el ámbito nacional**

#### **VERSAT**

El VERSAT Sarasola constituye un Software de Gestión Contable – Financiero Integrado, que automatiza prácticamente todas las actividades de Planificación, Control de los Recursos, la Contabilidad General y el Análisis Económico de cualquier tipo de Entidad en nuestro País o fuera de este.

El software se mantiene en una constante actualización, ha ido ganando en subsistemas y nuevas facilidades, solicitadas por los nuevos clientes o por nuevas regulaciones establecidas por los Organismos Rectores de las actividades Económicas, Financieras e Informáticas del País. En todas las provincias y municipios existen comerciales e implantadores del Sistema, que contribuyen a su distribución, actualización y la corrección de cualquier falla, de forma que el usuario se sienta seguro con el producto.

Los instaladores y su documentación están disponibles en varios sitios tanto del Minaz, como del MIC, pero con un tiempo limitado de uso. Una vez decidida su utilización definitiva debe gestionarse el pago a través de una de las empresas mencionadas anteriormente dedicadas a su comercialización. Se debe tratar de NO explotar el sistema sobre SQL 2008 o inferior en ninguna de sus ediciones pues es un gestor que no tiene soporte por Microsoft. Se recomienda crear las condiciones para el uso en una versión SQL Server 2017 Revisión 12 o superior para fortalecer la seguridad del sistema. Optimizado para entornos Windows Server 2016 y SQL Server 2017 Revisión 12 (DATAZUCAR, 2021).

#### **RODAS XXI**

El Sistema Integral Económico Administrativo RODAS XXI posibilita automatizar el funcionamiento de cualquier empresa o unidad presupuestada. Es un sistema en constante desarrollo que tiene en cuenta su opinión para perfeccionarse y le ofrece mejores soluciones que harán más viable y rápido su trabajo (Rodas, s. f.). Las licencias de uso que la empresa comercializa incluyen además la instalación del sistema, el entrenamiento inicial de sus operadores, y la posibilidad de servicios de postventa para mantenimientos o capacitación de nuevo personal (Rodríguez, 2019).

Tiene disponibles once módulos para controlar desde las nóminas de una entidad, hasta sus almacenes e inventarios, los recursos humanos y toda la gestión contable y

financiera, de costos, facturación, entre otras funciones básicas que cada cliente puede contratar de forma independiente o como un paquete completo (Rodríguez, 2019).

El módulo de Almacén de Rodas XXI permite tener un control detallado de las existencias de la entidad, la cuales se actualizan en el mismo momento que se registra un movimiento. Se pueden realizar todo tipo de operaciones de entradas y salidas de los almacenes, con facilidad, en el momento que se desee, generando de forma automática, el documento asociado al movimiento de que se trate, previa configuración del sistema para ello. Es posible controlar varios almacenes trabajando cada uno de forma independiente. El módulo destaca por su gran sencillez y facilidad para trabajar con él, lo que permite un rápido aprendizaje, acortando así el tiempo necesario para que nuevos usuarios saquen el máximo de las posibilidades que ofrece el módulo (Rodas, 2021).

### **Suite Zun**

La Suite ZUN es un Sistema de Gestión de entidades Hoteleras y Extra hoteleras, desarrollado por el Grupo de Electrónica para el Turismo (GET,2019).

Aplicable a Entidades del MINTUR y Otros Ministerios. Integrado por módulos interrelacionados, flexibles, parametrizables y adaptables a los requisitos de cualquier entidad, independientemente de su complejidad.

Funcionamiento en entorno cliente servidor, desarrollado sobre software propietario, emplea como gestor de base de Datos SQL 2008, corre sobre plataforma de Windows. Está integrado por varios subsistemas o módulos como (GET, 2019):

### **ZUNst**

- Módulo de Inventario que gestiona los almacenes y el control de los productos tanto en existencias como en valores. Realiza operaciones como compras, movimientos entre secciones, gastos, ventas, escandallos de elaboración, despieces, reversiones y otras como fijaciones de Inventario, cambios de código y rebajas automáticas desde los puntos de venta. Facilita las consultas y la obtención de un gran número de listados y resúmenes de múltiple interés. Se conecta con la Contabilidad (ZUNacc), garantizando la generación automática de comprobantes y otros reportes de interés y con el módulo de

ZUNpos permitiendo la rebaja automática de las existencias de los productos en los departamentos asociados, a partir de las fichas técnicas definidas.

### **SUITE FACSI+**

FACSI+ es un sistema de facturación de servicio, está realizado en Visual Basic 6.0 con las características de funcionalidad y manipulación comunes a todas las aplicaciones del entorno gráfico de Microsoft Windows. La suite esta desarrolla por los Servicios de Información del Transporte (Sitrans). El sistema FACSI+ está concebido para el trabajo en redes locales de pequeña y mediana magnitud (*Manual de usuario FACSI+*). Permite la facturación de contratos, ventas, cuentas por cobrar, etc; así como la gestión de inventarios, productos y servicios, a través de los sistemas FACSI +, FACSINV +, FACSI TRANSCAR + (SUITE FACSI+, 2019).

### **FACSINV +**

- **"Facsinv+"** es un sistema de facturación de productos con control de inventario. Está realizado con las características de funcionalidad y manipulación comunes a todas las aplicaciones del entorno gráfico de Microsoft Windows. Incluye una serie de características y funcionalidades que lo convierten en una herramienta confiable para su labor cotidiana. Este sistema es de gran utilidad para los comerciales y económicos, ya que le garantiza al usuario ahorro de tiempo y trabajo. Posee un módulo contable en cada una de las modalidades, enlaza automáticamente los comprobantes con el sistema contable, tiene control automático de los cobros y de los inventarios en todos los almacenes, ofrece garantía en el control y seguridad del trabajo, además de brindar una amplia variedad de reportes de salida y otras prestaciones.

### **Distra**

El Sistema de Gestión de Cadenas de Suministros desarrollado por la XETID, permite el control de los procesos que gestionan la cadena logística mostrando la información actualizada del estado de los productos y sus movimientos. Contiene módulos altamente configurables con posibilidad de ser integrados a sistemas externos, permitiendo una introducción gradual de los mismos en la empresa. Cuenta con una herramienta de análisis de información que permite la integridad de los datos, la trazabilidad de todas las operaciones realizadas y la obtención de reportes apoyando a la toma de decisiones (XETID, 2021).

Componentes del sistema:

- Módulo de Gestión de Activo Fijo.
- Módulo de Gestión de Almacenes (Distribución y facturación).
- Módulo de Punto de venta y compra.
- Módulo de Gestión de talleres.
- Módulo de Gestión del Transporte.

### 1.2.2 Estudio de sistemas homólogos existentes en el ámbito internacional

#### **EXACT**

**Exact Software** es una de las principales compañías europeas proveedoras de soluciones para la gestión financiera y contable, fue fundada en Delft, Holanda en 1984. Ahora, 35 años más tarde, *Exact* es el líder del mercado del software de gestión financiera y empresarial. Ofrece soluciones específicas para determinados sectores empresariales, que consiguen optimizar íntegramente la gestión de sus procesos de negocio. Más de 550.000 empresas y despachos profesionales ya utilizan su innovador software en la nube para gestionar sus números. El software utiliza la arquitectura de base de datos "One-X" y el acceso al mismo se realiza mediante una aplicación de escritorio (Exact, 2021).

El costo supera los dos millones dólares, solo por las licencias, consultarías e instalaciones, más otra cifra apreciable por los mantenimientos e iguales anuales (DATAZUCAR, 2021).

#### **Easy WMS**

*Easy WMS* es un software potente, robusto, versátil y flexible, con funcionalidades multipropietario, multisite y multilingüe, que simplifica y optimiza al máximo la gestión de un almacén, sea cual sea su tamaño y tipo. Controla, coordina y gestiona todos los movimientos, procesos y operativas, multiplicando la rentabilidad en todas las áreas: recepción, almacenaje, preparación de pedidos y expedición de órdenes de salida.

*Easy WMS* está disponible en tres niveles de funcionalidad (*PRO*, *ADVANCED & ENTERPRISE*), para ajustarse a las necesidades reales de cada empresa y reducir los costes de implementación (Mecalux, 2021).

## AHORA

El **Sistema de Gestión de Almacén (SGA)** de **AHORA** es un módulo más, dentro del ERP que ofrece funcionalidades avanzadas y una serie de procesos adicionales que permiten adaptarse al ritmo y crecimiento del negocio requerido.

El software de gestión de almacén de **AHORA** permite **gestionar en tiempo real los flujos de información** asociados a las operaciones logísticas del almacén, e integrar la gestión del mismo en los principales Sistemas de Información (AHORA SGA, 2021).

Características del producto:

- AHORA CRM notifica sobre las tareas vencidas, pendientes o a punto de realizar. También avisa si existen presupuestos a punto de caducar. AHORA CRM te mantiene informado en todo momento para que nunca pierdas una oportunidad de venta.
- Completa gestión documental en todos los procesos de la empresa. Define quién y cómo puede acceder a cada documento.
- Con AHORA ERP consigues una gestión multi-almacén que permite un control exhaustivo del stock y el histórico de movimientos de tus artículos.
- Gestiona los trasposos de mercancías entre tiendas o entre almacenes a través de un sencillo circuito de logística. Controla el stock de cada almacén y consúltalo cómodamente.
- Consulta el stock de los productos en vivo y realiza solicitudes de compra o trasposos entre almacenes.

Tabla 1 Resultado del estudio de Homólogos [Elaboración Propia]

Sistema	Licencia	Dominio de aplicación		Código abierto	Modular
		WEB	Escritorio		
<b>VERSAT</b>	Privada		Si	No	Si

<b>RODAS XXI</b>	Privada		Si	No	Si
<b>Suite Zun</b>	Privada		Si	No	Si
<b>SUITE FACSI+</b>	Privada		Si	No	Si
<b>Distra</b>	Privado	SI	SI	No	SI
<b>EXACT</b>	Privada		Si	No	No
<b>Easy WMS</b>	Privada	Si	Si	No	No
<b>AHORA</b>	Privada	Si	Si	No	Si

**Conclusión de los Sistemas de Gestión de Almacén (SGA)**

Luego de concluido el análisis sobre los sistemas de gestión de almacenes tanto a nivel internacional como a nivel nacional, se llega a la conclusión que ninguno de los sistemas analizados cumple con los requisitos necesarios para dar solución al problema de investigación. Los mismos cuenta con pago de licencias para ser usados o lograr tener todas sus funcionalidades, en cambio la propuesta a solución tiene como uno de sus requisitos ahorrar a los bufetes colectivos gastos en cuanto a licencias de software. Los programas están desarrollados en diferentes plataformas principalmente en Windows, limitando su uso, en la propuesta se desea un sistema totalmente web. En los sistemas internacionales, existen funcionalidades que aportan gran valor al sistema en cuestión. Al ser sistemas privativos y no exponer su código fuente, bloquean el derecho a la modificación de sus funcionalidades. Al no poder acceder a su código y hacer uso del mismo, no cumple así con la independendencia tecnológica por la cual está luchando Cuba y que aboga por el uso de software de código abierto. Con lo anteriormente planteado se evidencia la necesidad de desarrollar el sistema web para la gestión material e insumos en los ONBC, decrementando el valor en cuanto a tiempo y esfuerzo que actualmente se emplea en la planificación de los almacenes e inventarios.

**1.3 Metodología de desarrollo de software**

El proceso de desarrollo de software se apoya en el uso de diferentes herramientas y tecnologías, las cuales, unidas a la metodología seleccionada, conforman el ambiente de desarrollo de un sistema.

Una metodología de desarrollo de software se refiere al entorno que se usa para estructurar, planificar y controlar el proceso de desarrollo de un sistema informático. Tienen como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. Las metodologías para el desarrollo del software imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. No existe una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto, exigiéndose así que el proceso sea configurable. Las metodologías de desarrollo se clasifican en dos clases: las metodologías tradicionales o robustas y las ágiles o ligeras.

### **Metodología de desarrollo de software AUP versión UCI**

La Universidad de las Ciencias Informáticas (UCI) desarrolló una versión de la metodología de desarrollo de software AUP (Proceso Ágil Unificado), con el fin de crear una metodología que se adapte al ciclo de vida definido por la actividad productiva de la universidad. Esta versión decide mantener para el ciclo de vida de los proyectos la fase de Inicio, pero modificando el objetivo de la misma y se unifican las restantes fases de la metodología de desarrollo de software AUP en una sola, nombrada Ejecución y agregándose también una nueva fase llamada Cierre (SÁNCHEZ, 2015).

Tabla 2 Fases de la variación de AUP para la UCI [Elaboración propia]

Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases (Variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y

		costo y decidir si se ejecuta o no el proyecto.
Elaboración	Ejecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
Construcción		
Transición		
	Cierre	En esta fase se analizan tanto los resultados del proyecto como la ejecución y se realizan las actividades formales de cierre del proyecto.

La metodología AUP-UCI plantea tres formas de encapsular los requerimientos: Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requerimientos por proceso (DRP), que los mismo se agrupan en cuatro escenarios :

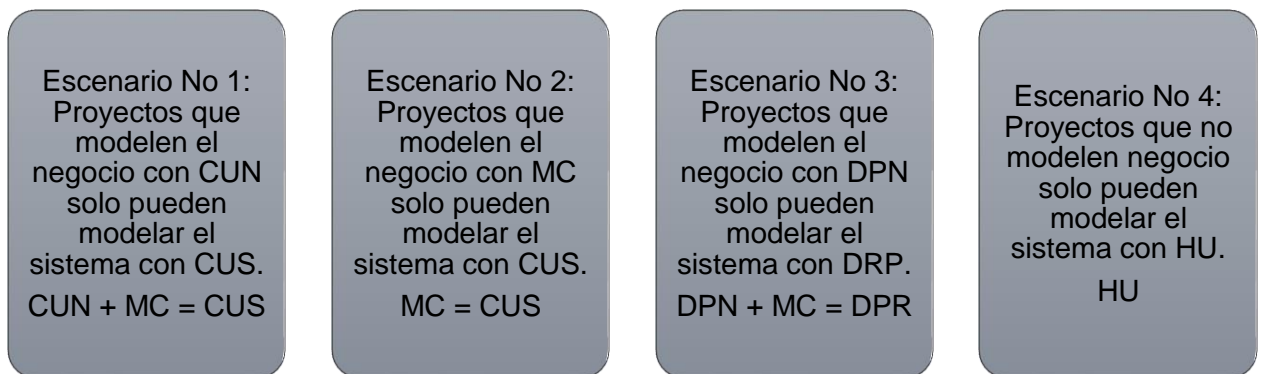


Figura 3 Escenarios de la metodología AUP-UCI [Elaboración propia]

Teniendo en cuenta los escenarios de la variación AUP-UCI, se decide encapsular los requisitos en el escenario dos. Para la decisión se analizó el criterio del escenario que plantea que: aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los



conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

#### **1.4 Tecnologías y herramientas para el desarrollo.**

Las herramientas informáticas son programas, aplicaciones o simplemente instrucciones usadas para efectuar tareas de modo más sencillo (Yanover, 2016). Con el objetivo de minimizar los costos, se propone utilizar tecnologías y herramientas que permitan su uso sin necesidad de pago de licencias.

##### **1.4.3. Lenguajes de Programación**

Un lenguaje de programación según Gervacio (2018) consiste en un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Estos suelen usarse para crear programas que controlen el comportamiento físico y lógico de una máquina y para expresar algoritmos con precisión. A continuación, se describen los lenguajes de programación a utilizar en el desarrollo de la propuesta de solución:

##### **Lenguaje del lado del cliente**

##### **Lenguaje *HyperText Markup Language* (HTML) 5**

Lenguaje de publicación especificado como un estándar por el W3C (*World Wide Web Consortium*) que permite la creación de páginas web. Inicialmente fue presentado por Tim Berners-Lee que propuso un sistema basado en hipertexto para el intercambio de información en la Web. La aparición del lenguaje influyó notablemente en el crecimiento de Internet, donde la información era distribuida mediante colecciones fragmentadas de textos, imágenes y sonidos. HTML es independiente de la plataforma utilizada y se basa fundamentalmente en el uso de etiquetas estructurales y semánticas, adecuadas para la creación de documentos relativamente simples que permiten simplificar su estructura (Jose, 2018).

Algunas de sus características son:

- Estructura del cuerpo: permite agrupar todas estas partes de una web en nuevas etiquetas que representarán cada una de las partes típicas de una página.

- Etiquetas para contenido específico: utiliza etiquetas específicas para cada tipo de contenido en particular, como audio, video, entre otros.
- Bases de datos locales: permite el uso de una base de datos local, con la que se puede trabajar en una página web por medio del cliente y a través de un API5.
- Aplicaciones web offline: permite la creación de aplicaciones web que funcionen sin necesidad de estar conectados a internet.

### **Lenguaje Cascading Style Sheets 3 (CSS)**

Hojas de Estilo en Cascada (del inglés *Cascading Style Sheets*) o CSS es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML (en-US) (incluyendo varios lenguajes basados en XML como SVG, MathML o XHTML). CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios. CSS es uno de los lenguajes base de la *Open Web* y posee una especificación estandarizada por parte del W3C (MOZILLA, 2021)

### **Bootstrap 4**

Bootstrap es un *framework* desarrollado por Twitter y de código abierto, orientado a *frontend* exclusivamente. Permite crear páginas web adaptables, que brindan a los desarrolladores de software la posibilidad de presentar el contenido de las páginas web en diferentes dispositivos sean de escritorio o móviles con un solo código fuente (Rodríguez Astudillo, 2018). Bootstrap posee plantillas predeterminadas, eliminando así la creación desde cero de un sitio web. Entre sus características más relevantes está la utilización de cajas, menús, botones, formularios, tipografías entre otros elementos; todos ellos basado en HTML y CSS con la ayuda de JavaScript.

### **JavaScript (JS)**

JavaScript es un lenguaje de programación ligero, interpretado, o compilado justo-a-tiempo (*just-in-time*) con funciones de primera clase. Si bien es más conocido como un lenguaje de scripting (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat. JavaScript es un lenguaje de programación basada en prototipos, multi-paradigma, de un solo hilo, dinámico, con soporte para programación orientada a objetos, imperativa y declarativa (por ejemplo, programación funcional) (MOZILLA, 2021)

### **Angular V 13.0.2**

Angular es una plataforma de desarrollo para crear aplicaciones de una sola página eficientes y sofisticadas. Construida sobre TypeScript, de código abierto y mantenida por Google. Los módulos, componentes y servicios son clases que utilizan decoradores. Estos decoradores marcan su tipo y proporcionan metadatos que le dicen a Angular cómo usarlos (*Angular*, 2021).

Como plataforma, Angular incluye:

- Un marco basado en componentes para crear aplicaciones web escalables
- Una colección de bibliotecas bien integradas que cubren una amplia variedad de características, que incluyen enrutamiento, administración de formularios, comunicación cliente-servidor y más.
- Un conjunto de herramientas para desarrolladores que lo ayudarán a desarrollar, compilar, probar y actualizar su código.

### **Lenguaje de lado del servidor**

#### **Lenguaje Hypertext Pre-processor (PHP) V 8.0.3**

Lenguaje de alto nivel con técnicas de Programación Orientada a Objetos, multiplataforma, sencillo de usar, rápido, integrable, además de ser de software libre. PHP es uno de los lenguajes de programación que permite programar scripts del lado del servidor, insertados dentro del código HTML, con una variedad de funciones y documentación. PHP es orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos. La programación en PHP es segura y confiable debido a que el código PHP es invisible al navegador y al cliente, el servidor es el encargado de ejecutar el código y enviar su resultado HTML al navegador (Castillo, 2006).

Entre las características por las cuales se selecciona este lenguaje se encuentran:

- Laravel está implementado en PHP.
- Multiplataforma.
- Presenta soporte para el SGBD de PostgreSQL.

## **Laravel V 8.0**

Laravel (basado en el lenguaje de programación PHP) es un *framework* para web con una sintaxis elegante y expresiva. Un *framework* web proporciona una estructura y un punto de partida para crear aplicaciones, lo que le permite concentrarse en crear algo sorprendente mientras nos preocupamos por los detalles.

Laravel proporciona una experiencia de desarrollador increíble al tiempo que proporciona funciones poderosas como la inyección de dependencias exhaustiva, una capa de abstracción de base de datos expresiva, colas y trabajos programados, pruebas de integración y unidad, etc. (Laravel, 2021).

## **Sistema gestor de base de datos**

### **PostgreSQL V 9.4**

PostgreSQL es un poderoso sistema de base de datos relacional de objetos de código abierto que usa y extiende el lenguaje SQL combinado con muchas características que almacenan y escalan de manera segura las cargas de trabajo de datos más complicadas. PostgreSQL se ha ganado una sólida reputación por su arquitectura probada, confiabilidad, integridad de datos, conjunto de características robustas, extensibilidad y la dedicación de la comunidad de código abierto.

PostgreSQL se ejecuta en todos los principales sistemas operativos, ha sido compatible con ACID desde 2001 y tiene complementos potentes como el popular extensor de base de datos geoespacial PostGIS. No es de extrañar que PostgreSQL se haya convertido en la base de datos relacional de código abierto elegida por muchas personas y organizaciones (PostgreSQL, 2021).

## **Servidor de aplicaciones web**

### **Apache V 2.4.46**

El proyecto del servidor HTTP Apache es un esfuerzo por desarrollar y mantener un servidor HTTP de código abierto para sistemas operativos modernos, incluidos UNIX y Windows. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que proporcione servicios HTTP en sincronía con los estándares HTTP actuales.

El servidor HTTP Apache ("httpd") se lanzó en 1995 y ha sido el servidor web más popular en Internet desde abril de 1996 (Apache, 2021).

## **Entorno de desarrollo**

### **Visual Studio Code V 1.62.3**

Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en su escritorio y está disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes (como C ++, C #, Java, Python, PHP, Go) y tiempos de ejecución (como .NET y Unity) (*VisualStudio Code*, 2021).

## **Lenguaje de modelado**

### **Lenguaje Unificado de Modelado (UML) V 2.0**

UML (Unified Modeling Language) fue adoptado como estándar del Object Management Group (Grupo Gestor de Objetos) en 1997 debido a que representa una colección de las mejores prácticas de ingeniería que han sido probadas con éxito en el modelado de sistemas. Es un lenguaje para la especificación, visualización, construcción y documentación de sistemas, no solo de software. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, pues ha sido diseñado para modelar cualquier tipo de soluciones informáticas, arquitectura o cualquier otra rama (Sarmiento, 2016).

## **Herramienta de modelado**

### ***Visual Paradigm V 8***

*Visual Paradigm for UML* es una herramienta CASE que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, implementación y pruebas. Ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite construir diagramas de diversos tipos, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (*Visual Paradigm.*, 2015).

## **Herramientas de validación**

### **Apache JMeter**

La aplicación Apache JMeter es un software de código abierto, desarrollada 100% en Java puro. Diseñada para cargar el comportamiento funcional de prueba y medir el

rendimiento. Originalmente fue diseñado para probar aplicaciones web, pero se ha expandido a otras funciones de prueba.

Apache JMeter se puede utilizar para probar el rendimiento tanto en recursos estáticos como dinámicos, aplicaciones web dinámicas.

Se puede utilizar para simular una carga pesada en un servidor, grupo de servidores, red u objeto para probar su fuerza o para analizar el rendimiento general bajo diferentes tipos de carga (JMeter, 2021).

### ***Acunetix Web Vulnerability Scanner***

Escáner de vulnerabilidades web o *Acunetix Acunetix Web Vulnerability Scanner (WVS)* es una herramienta automatizada de prueba de seguridad de aplicaciones web que audita sus aplicaciones web comprobando vulnerabilidades como inyecciones SQL, scripts de sitios cruzados y otras vulnerabilidades explotables de piratería. En general, *Acunetix WVS* escanea cualquier sitio web o aplicación web al que se puede acceder a través de un navegador web y utiliza el protocolo HTTP / HTTPS.

Además de escanear automáticamente en busca de vulnerabilidades explotables, WVS ofrece una solución sólida y única para analizar aplicaciones web estándar y personalizadas, incluidas las que se basan en scripts de cliente, como Aplicaciones web JavaScript, AJAX y Web 2.0.

*Acunetix WVS* es adecuado para organizaciones pequeñas, medianas y grandes con *intranets*, *extranets* y sitios web destinados a intercambiar o entregar información con el clientes, proveedores, empleados y otras partes interesadas (Acunetix, 2021).

## **1.5 Conclusiones del capítulo**

En este capítulo se han abordado los elementos teóricos que dan sustento a la propuesta de solución del problema planteado, en tal sentido se puede arribar a las siguientes conclusiones:

1. Se realizó un estudio sobre los principales conceptos asociados al dominio de la presente investigación y las relaciones entre ellos. Esto permitió alcanzar un mayor dominio sobre los fundamentos de los sistemas de gestión y aseguramiento material.
2. Se mantuvo una búsqueda de soluciones implementadas a nivel internacional y nacional que tuviesen relación con los sistemas de gestión y aseguramiento material,

pero no se encontró ningún sistema que cumpla con las soluciones al problema de investigación; sin embargo, como resultado de la búsqueda se obtuvieron características y funcionalidades claves para el desarrollo del sistema.

3. La metodología a utilizar a lo largo del sistema propuesto, en este caso AUP-UCI adaptándose al ciclo de vida de los proyectos de la universidad. Se arribó a la conclusión de la utilización del escenario dos de la metodología AUP-UCI, siendo esta la que mejor encapsula los requisitos de la propuesta de solución.

4. El análisis de la documentación de los SGA, sobre la metodología de desarrollo, así como las herramientas, tecnologías y lenguajes de programación utilizados en la implementación de los mismos, permiten especificar el ambiente de desarrollo para la propuesta de solución.

## **CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA DE GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”.**

### **2.1 Introducción**

En este capítulo, a partir del estudio de los procesos del negocio, sus descripciones y su modelado, se describe el sistema a desarrollar. Se obtienen los artefactos relacionados a la ingeniería de software aplicada a la propuesta de solución tomando como punto de partida el problema de investigación. Además, se plasman los requisitos funcionales y no funcionales de la propuesta, así como los diferentes artefactos relacionados con la metodología de desarrollo.

### **2.2 Descripción de la propuesta de solución**

La propuesta de solución a implementar surge principalmente por la necesidad de eliminar esfuerzos y tiempos en la consolidación anual en la ONBC. Desde hace años la entidad está haciendo uso de un documento Excel para llevar a cabo esta tarea. Con el nuevo empleo de las tecnologías y la gama de posibilidades que las misma brinda, se decidió llevar a cabo un sistema capaz de gestionar toda la información necesaria para la consolidación.



## CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”

El sistema debe cumplir con todos los requisitos funcionales y no funcionales que se describen en el epígrafe siguiente. Debe contar con características como, adicionar

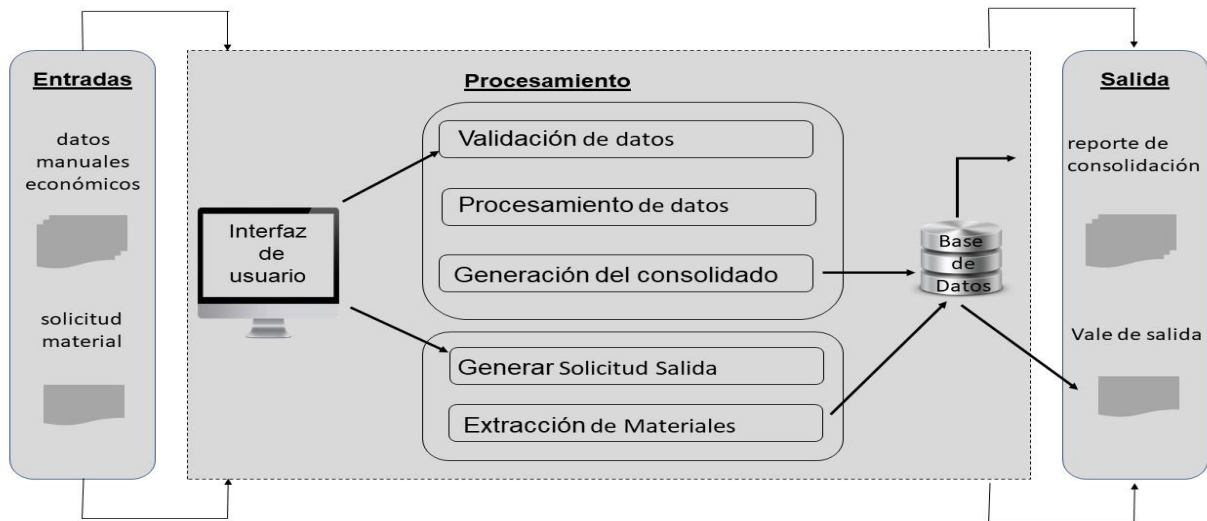


Figura 4 Descripción de la propuesta de solución [Elaboración propia]

usuarios al sistema con roles y permisos que posibiliten un correcto uso del sistema y mayor control de la información que se maneja; creación de categorías para los insumos nuevos y la edición de los datos de los mismos; gestionar los insumos según los diferentes conceptos como transporte, construcción e inversiones; gestionar los gastos por las áreas en las provincias; permitir general e imprimir un reporte general con los datos; además gestionar las solicitudes de materiales de los almacenes. Para lograr esto será desarrollado un sistema web, que brinda las facilidades de trabajar con esta información de forma fácil para los usuarios en general.

### 2.3 Modelo Conceptual

## CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”

Un modelo conceptual es un artefacto de la disciplina de análisis, construido con las reglas UML. Tiene como objetivo comprender y describir las clases más importantes, así como, identificar y explicar los conceptos significativos en el dominio del problema, identificando los atributos y las asociaciones existentes entre ellos (Sommerville,

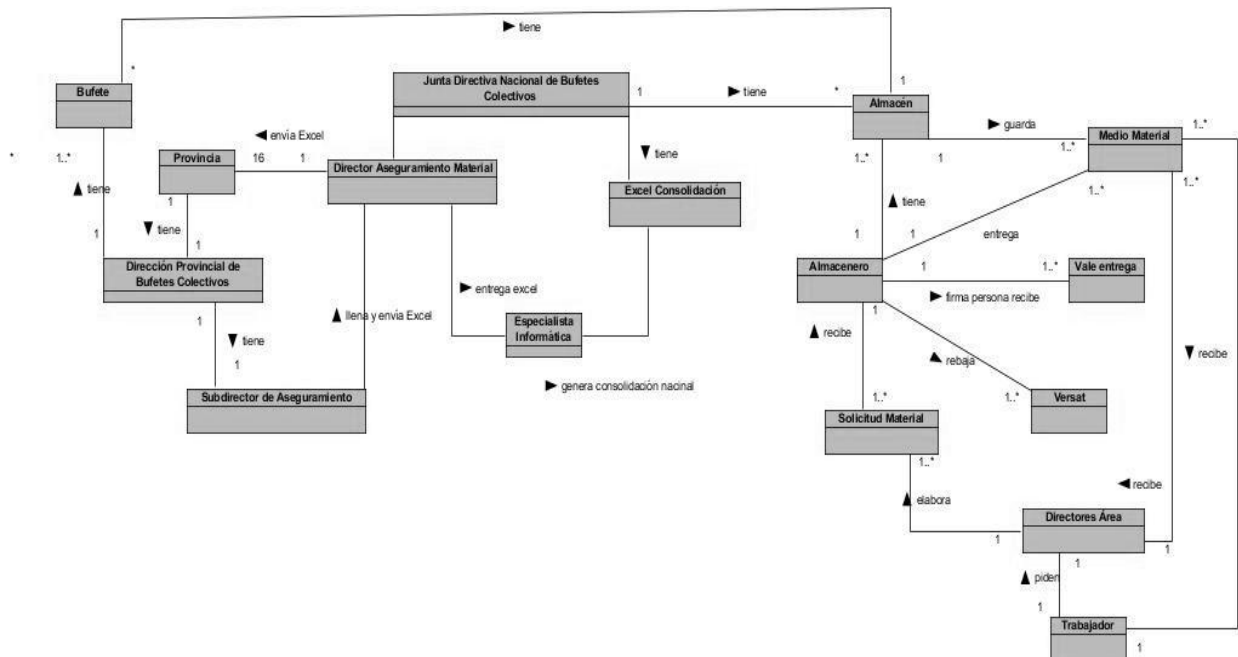


Diagrama 1 Modelo Conceptual [Elaboración propia]

2011).

### 2.4 Diagrama de caso de uso del sistema

Los diagramas de caso de uso son una técnica para capturar requisitos o información de cómo un sistema o negocio trabaja, y están compuesto por los casos de uso, los actores que se pueden definir como algo con comportamiento, como una persona (identificada por un rol), sistema informatizado u organización (Larman, 2003), y las relaciones existentes entre ambos.

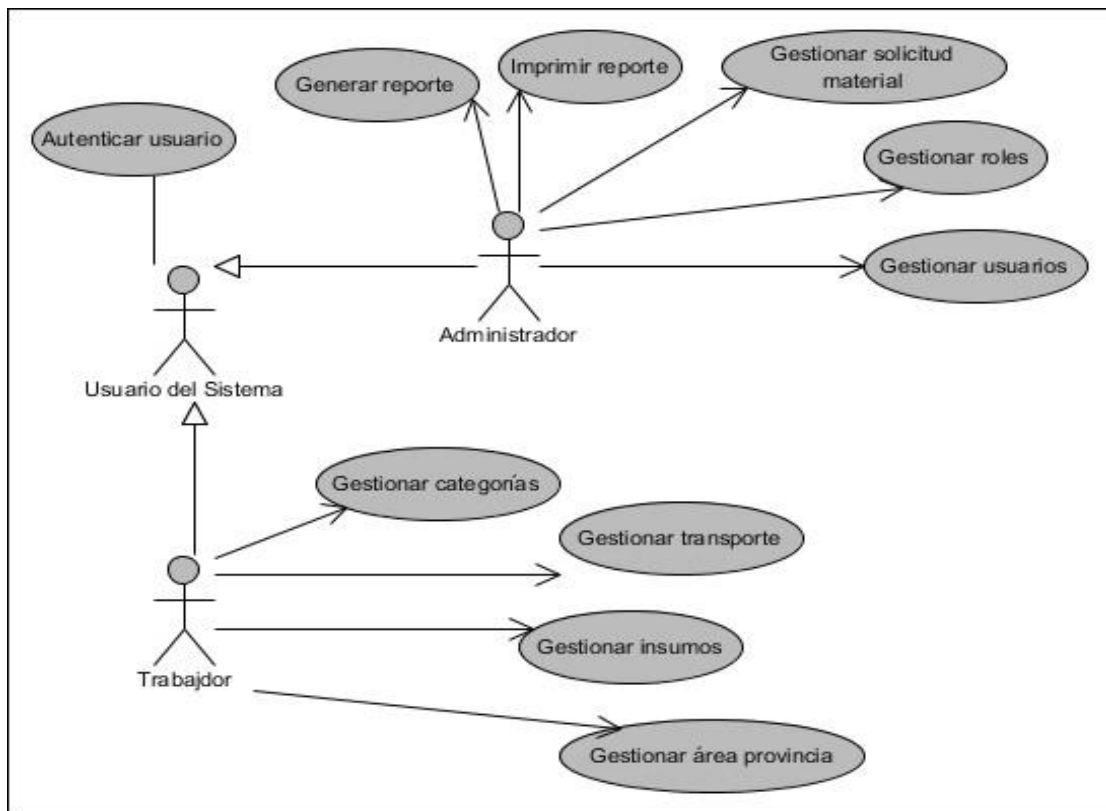


Diagrama 2 Diagrama de casos de uso del sistema [Elaboración propia]

## 2.5 Diagramas de clases del diseño

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces software en una aplicación. A diferencia de las clases conceptuales del Modelo del Dominio, las clases de diseño de los DCD muestran las definiciones de las clases software en lugar de los conceptos del mundo real (Sommerville, 2011).

## CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”

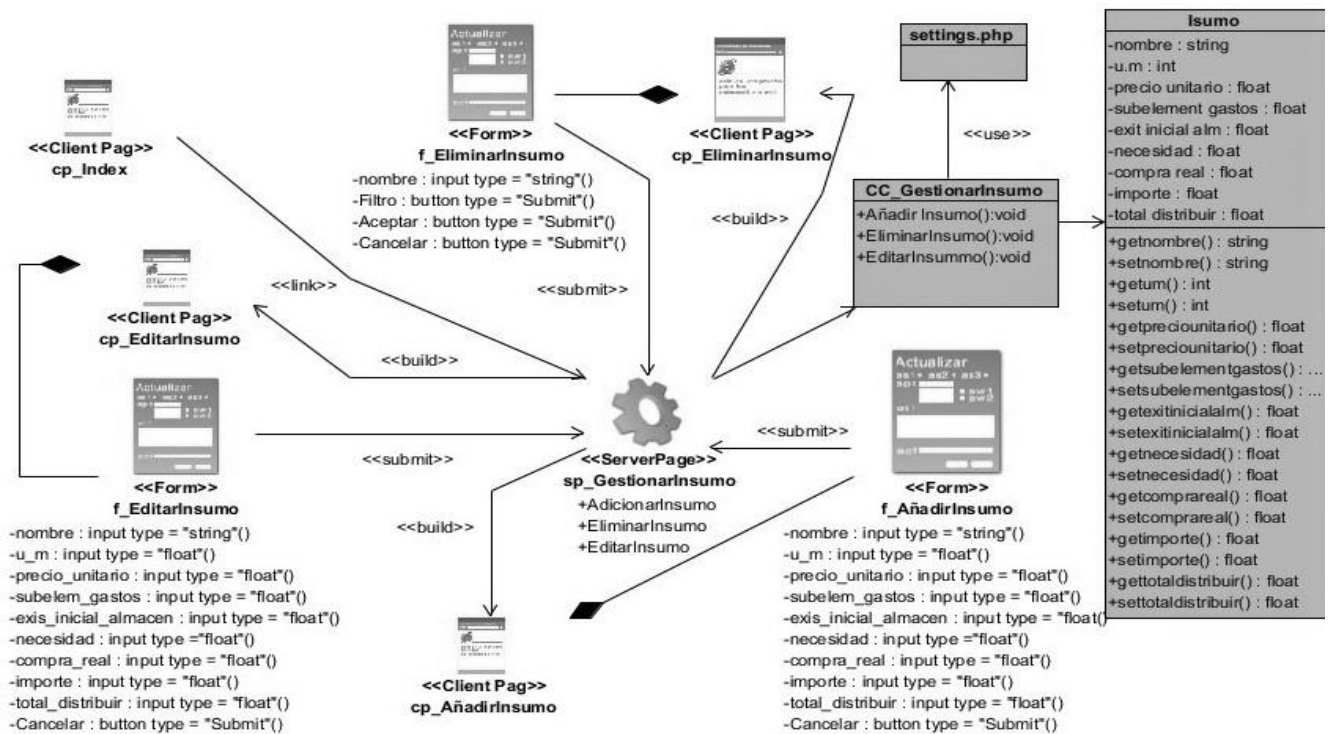


Diagrama 3 DCD con estereotipos web Gestionar Insumos [Elaboración propia]

El DCD está compuesto por 1 *server\_page* (página controladora/servidora) que se corresponde con la clase principal de gestión del contenido insumo, 4 *client\_page* (página cliente/vista) que muestran las funcionalidades añadir, editar, eliminar, y la página *index* correspondiente a la página principal del sistema, 3 formularios que contienen los campos, botones y filtros asociados a los contenidos, la clase controladora que contiene todas las funcionalidades de la *server\_page* (página controladora/servidora) , la clase que permite la conexión con la base de datos y la clase de la base de datos con los campos y funciones asociados al contenido insumo.

### 2.6 Diagrama de secuencia

Los diagramas de secuencia (DS) en el UML se usan principalmente para modelar las interacciones entre los actores y los objetos en un sistema, así como las interacciones entre los objetos en sí (Sommerville, 2011).

## CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”

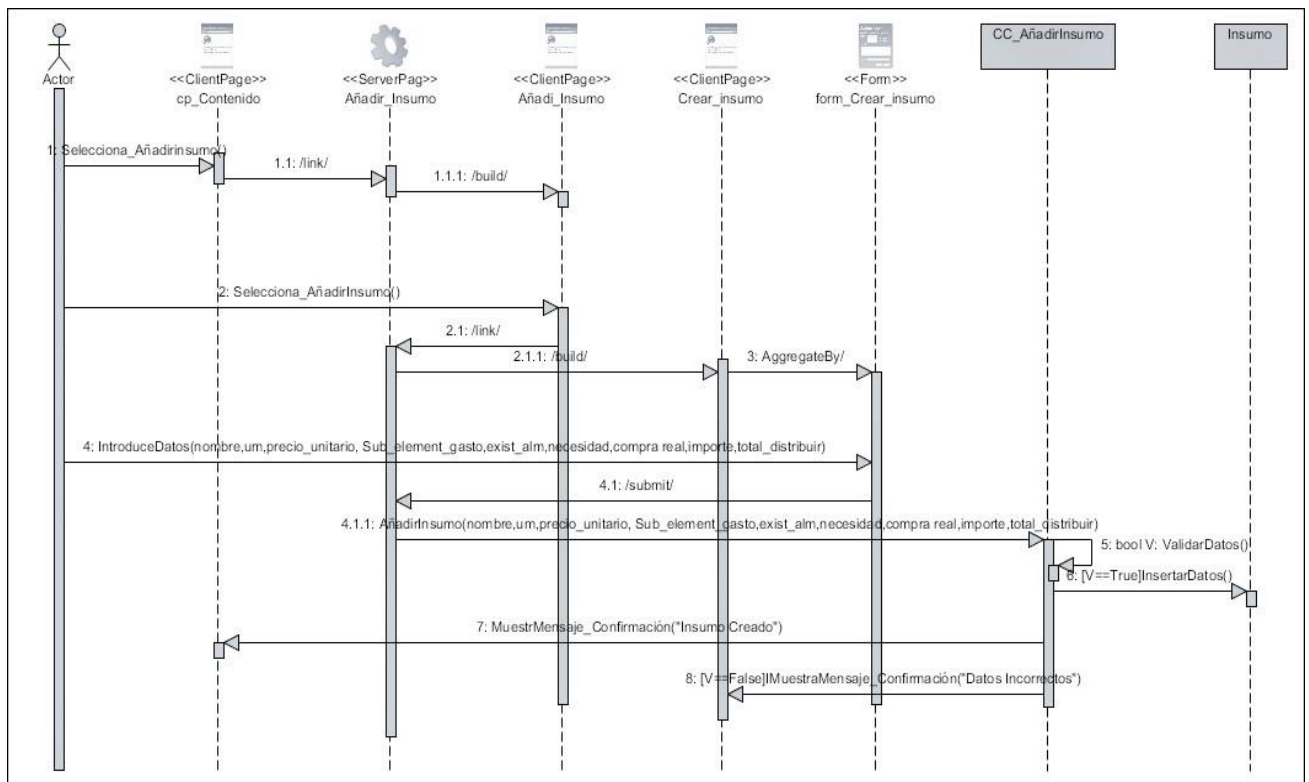


Diagrama 4 Diagrama de secuencia Crear Insumo [Elaboración propia]

El flujo representado en el DS comienza cuando el actor (administrador) del sistema selecciona en el mismo la opción de añadir insumo. La página cliente intermediaria hace la solicitud a la server\_page (página controladora/servidora). La página controladora construye la client\_page (página cliente/vista) que permite añadir el insumo, esta hace link a la controladora que crea la página cliente con el formulario con los campos asociados a al insumo. El actor introduce los datos que se envían a la server\_page (página controladora/servidora) y automáticamente se adiciona el insumo en la clase controladora donde se validan que los datos estén correctos. En caso de que los datos no hayan sido insertados de forma correcta se muestra un mensaje de error en la página cliente desde donde se creó la planificación, en el caso contrario se muestra un mensaje de confirmación en la clase desde donde el actor realizó la solicitud inicial.

### 2.7 Modelo de datos

Un modelo de base de datos muestra la estructura lógica de la base, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos, la relación que existe entre sí, los procesos que los transforman y cómo se accede a ellos. Se basa en

## CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”

la identificación de los objetos primarios que va a procesar el sistema, la composición y atributos de los mismos. En algunos casos, esta base de datos es independiente del sistema software (Sommerville, 2011).

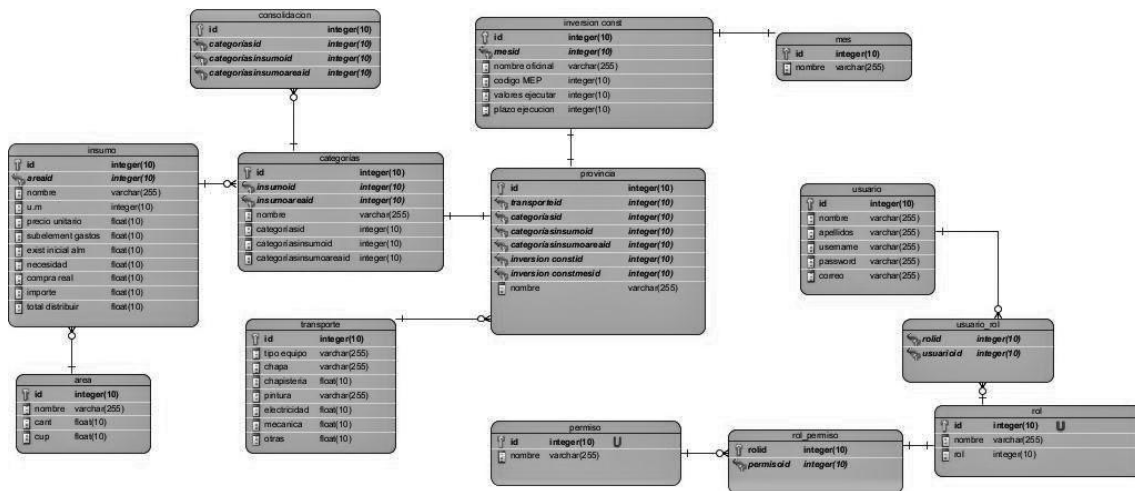


Diagrama 5 Modelo de datos [Elaboración propia]

### 2.8 Requisitos de la propuesta de solución

Los requerimientos para un software son las descripciones de lo que el sistema debe hacer, el servicio que ofrece y las restricciones en su operación. Tales requerimientos reflejan las necesidades de los clientes por un sistema que atiende cierto propósito (Sommerville, 2011).

#### 2.8.1 Requisitos funcionales

Los requisitos funcionales (RF) son declaraciones de las funcionalidades que debe cumplir el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (Pressman, 2002).

Tabla 3 Descripción de requisitos no funcionales [Elaboración propia]

Requisitos Funcionales	Nombre del requisito funcional	Prioridad	Complejidad
RF1	Añadir usuario del sistema	Alta	Baja
RF2	Editar usuario	Alta	Baja

CAPÍTULO 2: “ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS”

RF3	Eliminar usuario	Alta	Baja
RF4	Mostar datos del usuario	Alta	Baja
RF5	Listar Usuarios	Alta	Media
RF6	Autenticar usuarios	Alta	Media
RF7	Añadir roles	Alta	Baja
RF8	Editar roles	Alta	Baja
RF9	Eliminar roles	Alta	Baja
RF10	Cambiar asignación de permisos	Alta	Media
RF11	Añadir insumo	Alta	Baja
RF12	Editar insumo	Alta	Baja
RF13	Eliminar insumo	Alta	Baja
RF14	Añadir categoría de insumo	Alta	Baja
RF15	Editar categoría de insumo	Alta	Baja
RF16	Eliminar categoría de insumo	Alta	Baja
RF17	Añadir área provincial	Alta	Baja
RF18	Editar área provincial	Alta	Baja
RF19	Eliminar área provincial	Alta	Baja
RF20	Añadir equipo de transporte	Alta	Baja
RF21	Editar equipo de transporte	Alta	Baja
RF22	Listar equipo de transporte	Alta	Baja
RF23	Filtrar equipo de transporte	Alta	Media

RF24	Eliminar equipo de transporte	Alta	Baja
RF25	Añadir solicitud de materiales	Alta	Baja
RF26	Editar solicitud de materiales	Alta	Alta
RF27	Mostrar solicitudes de materiales	Alta	Alta
RF28	Exportar solicitud de materiales	Alta	Alta
RF29	Generar Reporte	Alta	Alta
RF30	Imprimir Reporte	Alta	Alta

### 2.8.2 Requisitos No Funcionales

Los requisitos no funcionales (RnF) hacen referencia a las propiedades emergentes del sistema: fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento, son limitaciones sobre servicios o funciones que ofrece el mismo. Los requisitos no funcionales se aplican al sistema como un todo, más que a características o a servicios individuales (Sommerville, 2011).

Tabla 4 Descripción de requisitos no funcionales [Elaboración propia]

N.º	Descripción
<b>Usabilidad</b>	
RNF1	El sistema para la gestión y aseguramiento material para la ONBC debe ser una aplicación web.
RNF2	La aplicación debe presentar una interfaz agradable e intuitiva.
<b>Confiabilidad</b>	
RNF3	La información manejada por el sistema está protegida de acceso no autorizado de usuarios, definiéndose los permisos según sus roles.
<b>Eficiencia</b>	



RNF4	El sistema debe permitir que los usuarios interactúen con él de manera concurrente.
RNF5	El tiempo de demora de una petición al servidor debe ser menor de cinco (5) segundos aproximadamente.
<b>Restricciones de Implementación y Diseño</b>	
RNF6	El sistema debe ser desarrollado en su totalidad con tecnologías de código abierto.
<b>Software</b>	
RNF7	Para el uso del sistema se requiere una PC cliente con cualquier sistema operativo, que se pueda instalar navegadores web para el uso de la aplicación.
RNF8	La comunicación entre la PC cliente y el servidor de aplicaciones web se realiza a través del protocolo HTTPS.
<b>Hardware</b>	
RNF9	Teniendo en cuenta que en la computadora va a ejecutarse un servidor web, se requiere como mínimo 80gb de disco duro, tarjeta de red de 100 MB, un procesador Pentium 4 y 4 GB de RAM para que la aplicación funcione correctamente.

## 2.9 Arquitectura de software

Los *frameworks* Angular y Laravel utilizan la arquitectura Modelo-Vista-Controlador (MVC). Es un patrón de diseño de software que utiliza y a la vez mantiene separados los componentes descritos en esta metodología. Además, es muy utilizado para la arquitectura en la mayoría de los frameworks modernos y uno de los más usados en la industria para poder crear proyectos escalables y modulares. Esta arquitectura se utiliza tanto en componentes gráficos básicos, como en sistemas empresariales.

La idea detrás de MVC es que cada uno de los componentes en su código tenga un propósito, y que esos propósitos sean diferentes. Además de que la forma en que se

relacionan estas partes ayuda con la ventaja de realizar un mejor mantenimiento en el futuro. Principalmente, tiene como objetivo dar soporte a los modelos funcionales y mapas mentales de la información relevante para los usuarios, permitiendo un modelo que facilite la consulta y manejo de los mismos. Este patrón permite una separación muy clara de los datos de la aplicación que consta de tres partes interconectadas: vista, modelo y controlador (Sánchez, 2020)

- **Vista:** este elemento hace referencia a la parte de una aplicación que considera la interfaz gráfica. Es decir, cada elemento gráfico que interactúa con el usuario forma parte de la Vista. Su función es obtener la información que requiere el usuario, lo cual se denomina como, evento. Esta capa siempre muestra la información proporcionada por el Modelo.
- **Modelo:** esta capa tiene la función de relacionar y gestionar los datos con los cuales la aplicación va a operar, como consultas, actualizaciones, creación de información o eliminación. Todo esto se le denomina como, Lógica de Negocio. En otras palabras, esta parte se refiere a la transformación de las actividades del mundo real a la forma en la que se va a modificar la información.
- **Controlador:** este componente responde ante eventos o acciones que realiza el usuario a través de la Vista para poder solicitar una operación de la información. Además, tiene la tarea de la elección de la Vista que se mostrará al usuario de acuerdo con la solicitud recibida, por lo que es el vínculo entre el Modelo y la Vista.

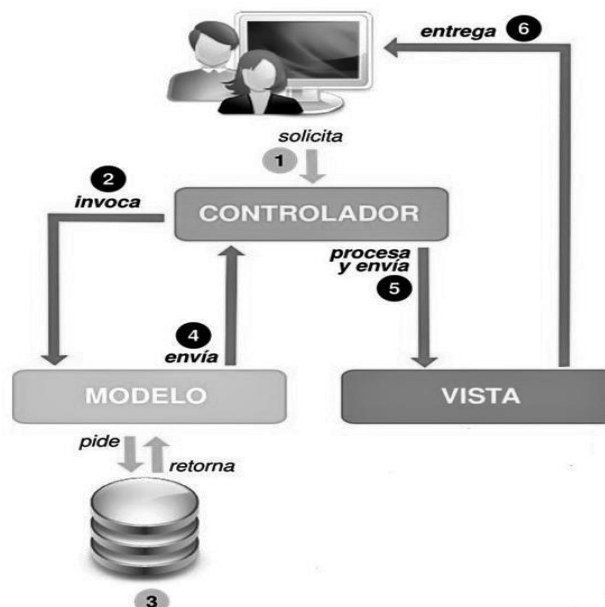


Figura 5 Arquitectura MVC [Elaboración propia]

## 2.10 Patrones de Diseño

Los patrones de diseño son una técnica para resolver problemas comunes en el desarrollo web y otros ámbitos referentes al diseño de interacción o interfaces. Establece una relación entre un determinado contexto, un problema y una solución. Para el diseño de software, el contexto permite al lector entender el ambiente en el que reside el problema y qué solución sería apropiada en dicho ambiente. Un conjunto de requerimientos, incluidas limitaciones y restricciones, actúan como sistema de fuerzas que influyen en la manera en la que puede interpretarse el problema en este contexto y en cómo podría aplicarse con eficacia la solución (Pressman, 2010).

Los Patrones Generales de Software para la Asignación de Responsabilidades (GRASP), fomentan una serie de buenas prácticas para el diseño de software. Se emplean para describir la asignación de responsabilidades a objetos. De esta familia, para el desarrollo del sistema se utilizaron los siguientes patrones:

- **Experto en Información:** Asigna una responsabilidad al experto en información, la clase que tiene la información necesaria para llevar a cabo la responsabilidad en la creación de objetos y métodos según el modelo de negocio.
- **Controlador:** Asigna la responsabilidad de gestionar un mensaje de un evento del sistema a una clase que represente una de estas opciones:
  - a. Representa el sistema global, dispositivo o un subsistema (controlador de fachada).
  - b. Representa un escenario de caso de uso en el que tiene lugar el evento del sistema (controlador de caso de uso o sesión)
- **Alta Cohesión:** Asigna responsabilidades de manera que la cohesión permanezca alta. La información contenida por las clases debe estar relacionadas con el objetivo de su creación.
- **Bajo Acoplamiento:** Asigne responsabilidades de manera que el acoplamiento (innecesario) se mantenga bajo. El propósito es mantener las clases lo más distante entre ellas, teniendo en cuenta que de ser necesario una edición en

algunas de la clase las demás no se vean afectadas, eliminando las dependencias innecesarias entre clases (Larman, 2003).

### **2.11 Conclusiones del capítulo**

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

- I. El análisis de las características del sistema y la modelación del dominio permitió identificar los principales requisitos funcionales y no funcionales del sistema de gestión y aseguramiento material, los cuales fueron agrupados y categorizados por casos de uso.
- II. El diseño de los diagramas de clases el entendimiento sobre la composición física y lógica del sistema.
- III. Los artefactos generados según la metodología de desarrollo utilizada y los patrones de arquitectura y diseño descritos, constituyeron una guía fundamental para la construcción de la propuesta de solución.

### CAPÍTULO 3: "IMPLEMENTACIÓN Y VALIDACIÓN DEL SISTEMA PARA LA DE GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS".

Con el objetivo de asegurar que el sistema de gestión propuesto funciona de acuerdo a los requisitos definidos y materializar en forma de componentes la fase de análisis y diseño, en el presente capítulo se proponen las estrategias de pruebas a utilizar para comprobar la calidad del software durante la etapa de validación.

#### 3.1 Diagrama de despliegue

Un modelo de despliegue es un diagrama estructurado que muestra la arquitectura del sistema desde el punto de vista de distribución de los artefactos del software en los destinos de despliegue; se definen los artefactos como representaciones de elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo (Sarmiento, 2016).

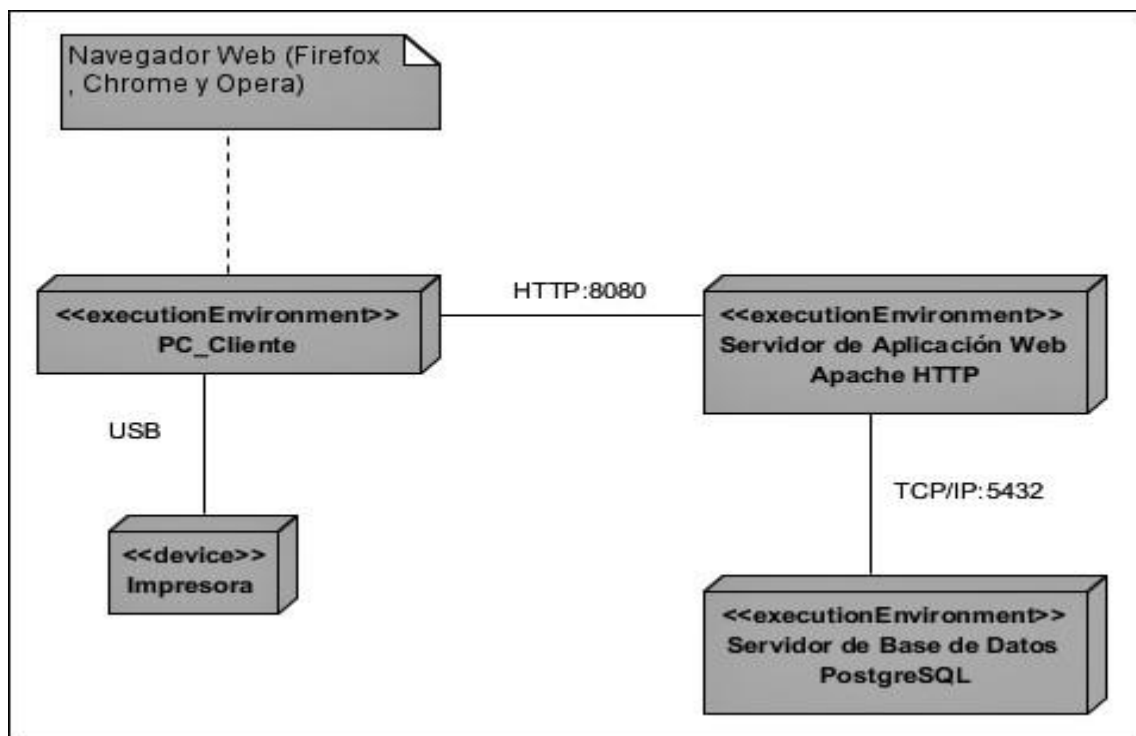


Diagrama 6 Diagrama de despliegue [Elaboración propia]

### **Descripción de elementos e interfaces de comunicación**

**Dispositivo PC\_Cliente:** La estación de trabajo necesita un navegador web para conectarse al sistema hospedado en el servidor de aplicaciones utilizando el protocolo de comunicación HTTP/HTTPS (8080/443).

**Servidor de aplicación web:** Es la estación de trabajo que hospeda el código fuente de la aplicación y que le brinda al usuario las interfaces para realizar los procesos del sistema. Esta estación se comunica con el servidor de base de datos donde se almacenan los datos de la aplicación realizando la comunicación mediante el protocolo TCP/IP (Transmission Control Protocol/Internet Protocol). Este servidor debe utilizar un sistema operativo cuyas propiedades mínimas sean un procesador Pentium 4 y 4GB de memoria RAM.

**Servidor de BD(PostgreSQL):** Este servidor es el encargado del almacenamiento de los datos del sistema. Se comunica con el servidor de aplicaciones del sistema mediante el protocolo TCP:5432, posibilitando el acceso mediante el usuario con privilegios para las operaciones determinadas a realizarse en el mismo.

**Dispositivo impresora:** Dispositivo periférico de salida del ordenador que permite producir una gama permanente de texto o gráficos de documentos almacenados en un formato electrónico, imprimiéndolos en medios físicos. Esta encargada de imprimir los reportes generados por el sistema una vez generada la consolidación a nivel nacional de la ONBC.

**Navegador Web:** Es un programa que permite ver la información que contienen una página web. El navegador interpreta el código; HTML generalmente, en el que está escrita una página web y lo presenta en pantalla permitiendo al usuario interactuar con su contenido y navegar.

### **3.2. Estándares de codificación**

Los estándares de codificación constituyen un principio esencial en el desarrollo de software. Garantizan que el código obtenido sea fácil de leer, entendido y modificado independientemente de quien haya sido el desarrollador del producto. Son una guía para el equipo de desarrollo, permiten asegurar que el código presente calidad y no contenga errores. PHP, como lenguaje de programación en el *backend*, proporciona a

sus desarrolladores un conjunto de normas para fomentar el código de una forma uniforme para todos.

A continuación, se proponen reglas a seguir en la implementación de la propuesta de solución para la gestión y aseguramiento material de la ONBC. Estas reglas definen los estándares de codificación para el lenguaje PHP, propuesto como una de las tecnologías empleadas en la solución.

### **Etiquetas de apertura y cierre**

Se debe utilizar las etiquetas `<?php` y `?>`, y en ningún caso la versión corta `<? y ?>`. En general se omite la etiqueta de cierre de PHP (`?>`) al final de los archivos `.module` y `.inc`. Esta convención evita que se puedan quedar olvidados espacios no deseados al final del archivo (después de la etiqueta de cierre `?>`), que serían identificados como salida HTML y podrían provocar un error muy típico, "Cannot modify header information-headers already sent by...". Por tanto, la etiqueta de cierre final del archivo (`?>`) es opcional (Gil, 2012).

```
<?php
echo "Hola mundo";

// ... más código

echo "Última sentencia";

// el script finaliza aquí sin etiqueta de cierre de PHP
```

Figura 6 Ejemplo de etiquetas de apertura y cierre (PHP: Hypertext Preprocessor, 2021)

### **Indentación**

La indentación consiste en insertar espacios en blanco o tabuladores en determinadas líneas de código para facilitar su comprensión. En programación se emplea indentación para anidar elementos. Se debe indentar con 2 espacios, nunca con tabuladores. Además, no se debe dejar espacios en blanco al final de cada línea (Gil, 2012).

```
<?php
    echo 'Esto es una prueba';
?>

<?php echo 'Esto es una prueba' ?>

<?php echo 'Hemos omitido la última etiqueta de cierre';
```

Figura 7 Ejemplo de indentación (PHP: Hypertext Preprocessor, 2021)

### Operadores

Los operadores binarios, que se utilizan entre dos valores, deben separarse de estos valores, a ambos lados del operador, por un espacio. Por ejemplo, `$max_age = 452`, en el lugar de `$max_age=452`. Esto se aplica a operadores como `+`, `-`, `*`, `/`, `=`, `!=`, `>`, `<`, `.` (Concatenación de cadenas), `.=`, `+=`, `-=`. Los operadores unarios como `++`, `--` no deben tener separación (Gil, 2012).

```
<?php
$a = 1;
echo $a + $a++; // podría mostrar 2 o 3

$i = 1;
$array[$i] = $i++; // podría establecer el índice a 1 o 2
?>
```

Figura 8 Ejemplo de operadores (PHP: Hypertext Preprocessor, 2021).

### Uso de comillas

Se utilizan tanto comillas simples como la ('cadena') como las comillas dobles ("cadena") para delimitar las cadenas de caracteres. Las comillas dobles son necesarias si se desean incluir variables dentro de las cadenas de texto (Gil, 2012).



```
<?php
$x = 4;
// esta línea podría resultar en una salida inesperada:
echo "x menos uno igual a " . $x-1 . ", o eso espero\n";
// ya que se evalúa como esta línea:
echo (("x menos uno igual a " . $x) - 1) . ", o eso espero\n";
// la precedencia deseada se puede forzar con paréntesis:
echo "x menos uno igual a " . ($x-1) . ", o eso espero\n";
?>
```

Figura 9 Ejemplo de comillas (PHP: Hypertext Preprocessor, 2021)

### Uso de punto y coma (;) en código PHP

Aunque PHP permite escribir líneas de código individuales sin el terminador de línea (;), como por ejemplo `<?php print $title ?>`. Es siempre obligatorio: `<?php print $title;?>`. Es importante señalar que el cierre de la etiqueta php es opcional (Gil, 2012).

```
<?php
function doble($i)
{
    return $i*2;
}
$b = $a = 5;          /* asignar el valor cinco a la variable $a y $b */
$c = $a++;           /* post-incremento, asignar el valor original de $a
                    (5) a $c */
$e = $d = ++$b;      /* pre-incremento, asignar el valor incrementado de
                    $b (6) a $d y $e */
```

Figura 10 Ejemplo del uso del (;) (PHP: Hypertext Preprocessor,2021).

### Estructuras de control

Según (Gil, 2012) con respecto a las estructuras de control, hay que tener en cuenta las siguientes normas:

- Debe haber un espacio entre el comando que define la estructura (*if*, *while*, *for*) y el paréntesis de apertura. Esto es así para no confundir las estructuras de control con la nomenclatura de las funciones.

```
<?php
/* ejemplo 1 */

$i = 1;
while ($i <= 10) {
    echo $i++; /* el valor presentado sería
                $i antes del incremento
                (post-incremento) */
}

/* ejemplo 2 */

$i = 1;
while ($i <= 10):
    echo $i;
    $i++;
endwhile;
?>
```

Figura 11 Ejemplo de while (PHP: Hypertext Preprocessor, 2021).

- La llave de apertura ( { ) se situará en la misma línea que la definición de la estructura, separada por un espacio.

```
<?php
if ($a > $b) {
    echo "a es mayor que b";
}
?>
```

Figura 12 Ejemplo de if (PHP: Hypertext Preprocessor, 2021).

- Se recomienda usar siempre las llaves { } aun en los casos en que no sea obligatorio su uso (una sola "línea" de código dentro de la estructura de control).

```
<?php
while( ++$i < 10 );

echo $i; // 10

?>
```

Figura 13 Ejemplo de casos en que no es obligatorio el uso de ( {} ) (PHP: Hypertext Preprocessor, 2021)

- Las estructuras *else* y *elseif* se escribirán en la línea siguiente al cierre de la sentencia anterior.

```
<?php
$t = date("H");

if ($t < "10") {
    echo "Have a good morning!";
} elseif ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

Figura 14 Ejemplo de elseif ( PHP: Hypertext Preprocessor, 2021).

## Funciones

Los nombres de las funciones deben estar escritos en minúsculas y las palabras separadas por guión bajo. Además, se debe incluir siempre como prefijo el nombre del módulo o tema, para evitar así duplicidad de funciones. En su declaración, después del nombre de la función, el paréntesis de inicio de los argumentos debe ir sin espacio. Cada argumento debe ir separado por un espacio, después de la coma del argumento anterior (Gil, 2012).

```
<?php
function foo($arg_1, $arg_2, /* ..., */ $arg_n)
{
    echo "Función de ejemplo.\n";
    return $valor_devuelto;
}
?>
```

Figura 15 Ejemplo de funciones ( PHP: Hypertext Preprocessor, 2021).

### Arreglos (*Arrays*)

Los valores dentro de un array (o matriz) se deben separar por un espacio (después de la coma que los separa). El operador => debe separarse por un espacio a ambos lados. Cuando la línea de declaración del array supera los 80 caracteres, cada elemento se debe escribir en una única línea. En este último caso, la coma de separación del último elemento también se escribirá, aunque no existan más elementos. De esta forma se evitan errores al añadir nuevos elementos al vector (Gil, 2012).

```
<?php
$fruits = array (
    "frutas" => array("a" => "naranja", "b" => "plátano", "c" => "manzana"),
    "números" => array(1, 2, 3, 4, 5, 6),
    "hoyos" => array("primero", 5 => "segundo", "tercero")
);
?>
```

Figura 16 Ejemplo de array (PHP: Hypertext Preprocessor,2021).

### Nombres de archivos

Los nombres de archivos deben escribirse siempre en minúscula. La única excepción son los archivos de documentación, que tendrán extensión .txt y el nombre en mayúscula. Por ejemplo README.txt.

### Comentar el código

Para la realización de comentarios suelen emplear `/*` para comentarios en varias líneas y `//` para comentarios de una única línea. Se deben escribir frases completas, comenzándolas con mayúscula y terminándolas con un punto. En caso de que en el comentario se haga referencia a una constante, esta deberá escribirse en mayúscula (por ejemplo: `TRUE` o `FALSE`) (Gil, 2012).

```
$b = $a = 5;      /* asignar el valor cinco a la variable $a y $b */
$c = $a++;       /* post-incremento, asignar el valor original de $a
                  (5) a $c */
$e = $d = ++$b;  /* pre-incremento, asignar el valor incrementado de
                  $b (6) a $d y $e */
```

Figura 17 Ejemplo de comentarios en el código (PHP: Hypertext Preprocessor, 2021)

### 3.3 Estrategia de Pruebas

Para garantizar el completo funcionamiento de toda aplicación informática es necesario realizar pruebas para su validación y de esta forma evitar cualquier problema o insatisfacción por parte de los clientes. Estas pruebas se trazan mediante estrategias que permitan evaluar todos los aspectos de determinado producto teniendo en cuenta su característica y así de una forma u otra estar completamente confiados y poder garantizar el éxito del mismo (Rosabal, 2005).

Para verificar el cumplimiento de los RnF descritos fue diseñada una lista de chequeo [Anexo A], identificándose indicadores a chequear para la evaluación de cada requisito. En el caso de los requisitos de eficiencia las pruebas se realizarán en el entorno real de la aplicación por lo que no se mostrarán resultados de estas como parte de la investigación.

En el desarrollo de las pruebas se propone que intervengan los siguientes roles:

- Coordinador de la prueba
- Probador
- Jefe de equipo de desarrollo
- Desarrollador

CAPITULO 3: "IMPLEMENTACIÓN Y VALIDACIÓN DEL SISTEMA PARA LA GESTIÓN Y ASEGURAMIENTO MATERIAL DE LA ORGANIZACIÓN NACIONAL DE BUFETES COLECTIVOS".

---

En la tabla 5 puede observarse la propuesta de cronograma a planificar en el que se recogen las especificaciones de cada actividad del proceso de prueba.

Tabla 5 Cronograma de planificación para el proceso de pruebas: [Elaboración propia]

<b>Actividad</b>	<b>Fecha</b>	<b>Responsable</b>	<b>Participantes</b>
Elaboración de la estrategia de prueba	Día 1	Coordinador de la prueba	Coordinador de la prueba
Diseñar la prueba	Día 1	Coordinador de la prueba	Coordinador de la prueba
Montaje del entorno de prueba	Día 2	Coordinador de la prueba	Coordinador de la prueba
Primera iteración de las pruebas	Día 2	Coordinador de la prueba	Probador
Solución de los defectos en los artefactos en prueba y actualización de los CP y artefactos de apoyo	Día 3	Jefe de equipo de desarrollo	Desarrollador
Segunda iteración de las pruebas	Día 4	Coordinador de la prueba	Probador
Evaluación de los resultados de las pruebas	Día 5	Coordinador de la prueba	Probador

**Métodos de caja blanca:** se centran en la estructura de control del programa. Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada componente, ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa, ejecuten todos los bucles en sus límites y

con sus límites operacionales y ejerciten las estructuras internas de datos para asegurar su validez (Pressman, 2002).

**Métodos de caja negra:** son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno de un programa. La prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca; se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los métodos de caja blanca como son errores de funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. Las técnicas de prueba de caja negra se centran en el ámbito de información de un programa, de forma que se proporcione una cobertura completa de la prueba (Pressman, 2002).

#### **Pruebas unitarias**

Las pruebas unitarias son el proceso de probar componentes del programa tales como métodos o clases de objetos. Las funciones o los métodos individuales son el tipo más simple de componente. Las pruebas deben llamarse para dichas rutinas con diferentes parámetros de entrada (Sommerville, 2011).

Un método que podría ser aplicado para esta prueba sería el de caja blanca, donde las pruebas se enfocan en la estructura de control del programa. Los casos de prueba se derivan para asegurar que todos los enunciados en el programa se ejecutaron al menos una vez durante las pruebas y que todas las condiciones lógicas se revisaron (Pressman 2010).

#### **Pruebas funcionales**

Se denominan pruebas funcionales a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados (Rivasi, 2009).

A continuación se detallan algunos casos de prueba:

Tabla 5: Casos de prueba: Añadir usuario [ Elaboración propia]

<b>Caso de prueba: Añadir usuario</b>
<b>Nombre de requisito: Añadir usuario</b>
<b>Nombre de la persona que realiza la prueba:</b> Maykol Daniel González Matos
<b>Descripción de la prueba:</b> Prueba a la funcionalidad añadir usuario
<b>Entrada/Pasos de ejecución:</b> Se introducen los siguientes datos para añadir un usuario
<p><b><u>Datos:</u></b></p> <p><b>Nombre:</b> Milena</p> <p><b>Apellido:</b> Pérez Corcho Pérez</p> <p><b>Email:</b> <a href="mailto:milena@gmail.com">milena@gmail.com</a></p> <p><b>Contraseña:</b> 12345678</p> <p>El administrador presiona el botón Aceptar y si los datos están correctos se crea el usuario que podrá ser visualizado desde la vista (Listar Usuario), si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.</p>

Tabla 6: Casos de prueba: Editar usuario [ Elaboración propia]

<b>Caso de prueba: Editar usuario</b>
<b>Nombre de requisito: Editar usuario</b>
<b>Nombre de la persona que realiza la prueba:</b> Maykol Daniel González Matos
<b>Descripción de la prueba:</b> Prueba a la funcionalidad editar usuario
<b>Entrada/Pasos de ejecución:</b> Se modifican los siguientes datos para editar un usuario
<p><b><u>Datos actuales:</u></b></p> <p><b>Nombre:</b> Mln</p> <p><b>Apellido:</b> Pérez Corcho Pérez</p>



Email: [milena@mgail.com](mailto:milena@mgail.com)

**Datos después de editados:**

**Nombre:** Milena

**Apellido:** Pérez Corcho Pérez

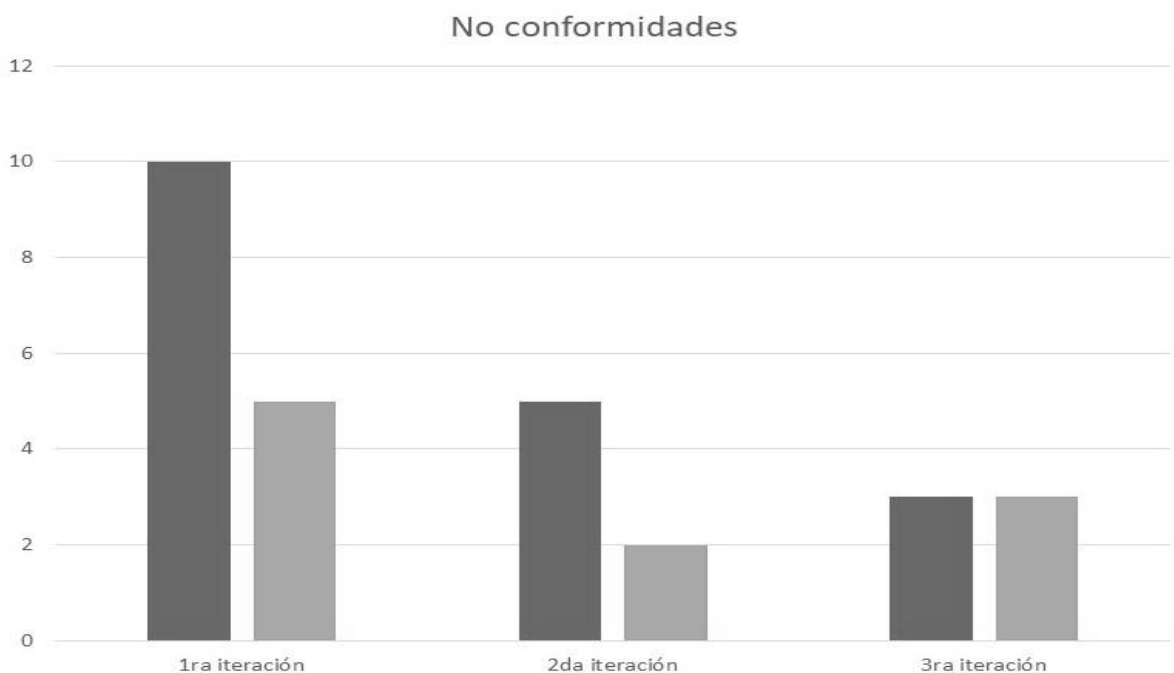
**Email:** [milena@mgail.com](mailto:milena@mgail.com)

El administrador presiona el botón Aceptar y si los datos están correctos actualiza el usuario que podrá ser visualizado desde la vista (Listar Usuario), si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.

**Resultado de las pruebas funcionales**

Para validar que el sistema cumpla con las funciones específicas para las cuales ha sido creado se realizaron las pruebas funcionales teniendo en cuenta los casos de prueba anteriores, arrojando los siguientes resultados: en una primera iteración 15 no conformidades de las cuales 5 fueron resueltas, en la segunda iteración quedaron pendientes 10 de las cuales se resolvieron 4 y en la última iteración, de las 6 restantes, se resolvieron todas. En la figura 17 se muestran los resultados obtenidos en la iteración de pruebas realizadas a parte del sistema de gestión y aseguramiento material.

Figura 18 Resultados de las iteraciones de las pruebas funcionales al gestionar usuario



Las no conformidades de funcionalidad, estuvieron relacionadas con diversas acciones. Por ejemplo, añadir un usuario en el sistema; todos los campos de datos en un primer instante no se almacenaban en la base de datos. Esta no conformidad fue resuelta con la inclusión de los campos en las tablas de la base de datos. En un segundo caso, al autenticarse los usuarios las contraseñas eran enviadas en texto plano, posibilitando la captura de las mismas en la red. Esta no conformidad fue resuelta con el encriptamiento de las contraseñas de los usuarios.

Las no conformidades de validación fueron encontradas en el formulario para crear un usuario, donde inicialmente el sistema permitía crear los usuarios con el mismo email de forma repetida, además permitía introducir caracteres extraños y no la estructura que el mismo debe seguir. La solución para los caracteres extraños fue validar la estructura del correo. El sistema muestra un mensaje de error si se intenta realizar esta acción de manera incorrecta.

Luego de concluir el sistema, se propone realizar las siguientes pruebas de las cuáles se enuncia su significado.

### **Pruebas de aceptación**

En ingeniería de software y pruebas de software, las pruebas de aceptación pertenecen a las últimas etapas previas a la liberación en firme de versiones nuevas a fin de determinar si cumplen con las necesidades y/o requerimientos de las empresas y sus usuarios (PRESSMAN, 2002).

Por su parte, la Junta Internacional de Cualificaciones de Pruebas de Software (ISTQB por sus siglas en inglés) define la "Aceptación" aplicado a la rama de la informática como: Pruebas formales que se realizan atendiendo las necesidades del cliente, requerimientos y procesos de negocio, realizadas para determinar si un sistema satisface los requerimientos de aceptación que permitan que el cliente pueda determinar si acepta o no el sistema (Herrerros, 2015).

Para realizar la prueba de aceptación, se deberá entregar la aplicación al cliente, el cual emite su criterio a través de una carta de aceptación, con sus consideraciones respecto a las ventajas que ofrecen el sistema y las necesidades que resuelve.

### **Pruebas de integración**

Las pruebas de integración son una técnica sistemática para construir la arquitectura de un software a la vez que se aplican las pruebas para encontrar errores asociados a las interfaces (Pressman, 2002). Además, comprueban que los componentes integrados realmente funcionan juntos, que sean llamados correctamente y que transfieran los datos correctos en el tiempo preciso a través de sus interfaces. Existen dos tipos de integración: incremental y no incremental. La integración no incremental consiste en combinar los componentes como un todo y probar el sistema, lo que arroja un gran número de errores. La integración incremental se fundamenta en dividir la integración en pequeños incrementos (módulos) y realizar la integración por separado. Así es más fácil aislar y corregir los errores.

Esta prueba se utilizaría por ejemplo, para verificar la integración del sistema con el servidor LDAP.

### **Pruebas de sistema**

Las pruebas funcionales tienen por objetivo probar que los sistemas desarrollados cumplen con las funciones específicas para los que han sido creados. Se utiliza el método de caja negra para evaluar funcionalmente la solución donde los probadores se enfocan en el funcionamiento de la interfaz del sistema a partir del estudio de sus entradas y salidas (Pressman, 2002). Para confeccionar los casos de prueba de caja negra existen distintas técnicas entre las que se encuentra la técnica de particiones equivalentes. Se basa en identificar las particiones para un sistema o componente. Las particiones de equivalencia son un conjunto de datos segregados por su validez en el sistema, donde cada uno de los miembros de los conjuntos debería ser procesado equivalentemente.

### **Pruebas no funcionales**

Una prueba no funcional es una prueba cuyo objetivo es la verificación de un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema (requisitos no funcionales)

### **Pruebas de rendimiento (carga y estrés)**

Las pruebas de rendimiento se diseñan para asegurar que el sistema pueda procesar su carga esperada. Éstas se ocupan tanto de demostrar que el sistema satisface sus

requerimientos, como de descubrir problemas y defectos en el sistema (Sommerville, 2011).

Las pruebas de carga consisten en simular una carga de trabajo similar y superior a la que tendrá cuando el sistema esté funcionando, con el fin de detectar si el software instalado (programas y aplicaciones) cumple con los requerimientos de muchos usuarios simultáneos y también si el hardware (servidor y el equipamiento computacional de redes y enlace que lo conecta a Internet) es capaz de soportar la cantidad de visitas esperadas (PRESSMAN, 2002).

Las pruebas de estrés evalúan la robustez y la confiabilidad del software sometándolo a condiciones de uso extremas. Entre estas condiciones se incluyen el envío excesivo de peticiones y la ejecución en condiciones de hardware limitadas. El objetivo es saturar el programa hasta un punto de quiebre donde aparezcan defectos potencialmente peligrosos (PRESSMAN, 2002).

### **Pruebas de rendimiento**

Para las pruebas de rendimiento se propone utilizar el software Apache Jmeter. Para ello se definen tener en cuenta las siguientes propiedades de las PC implicadas.

#### **Hardware de prueba (PC servidor)**

- Microprocesador: Intel(R) v 4
- Memoria RAM: 4.00 GB
- Disco Duro: 80 GB
- Tarjeta red: 100 MB/S

Definidos los requisitos de hardware se procede a configurar el Apache JMeter con los parámetros necesarios para lograr simular un total de 100 usuarios conectados simultáneamente, se realizan peticiones a diferentes partes del sistema para la gestión y aseguramiento material.

Se describen las variables que miden el resultado de las pruebas de carga y estrés:

- **Usuarios:** total de usuarios.
- **# Muestras:** El número de peticiones.
- **Media:** El tiempo medio transcurrido en milisegundos para un conjunto de resultados.

- **Mín:** El mínimo tiempo transcurrido en milisegundos para las muestras de la URL dada.
- **Máx:** El máximo tiempo transcurrido en un milisegundo para las muestras de la URL dada.
- **% Error:** Porcentaje de las peticiones con errores.
- **Rendimiento:** Rendimiento medido en base a peticiones por segundo/minuto/hora.
- **Kb/s Recibidos:** Rendimiento medido en Kbytes por segundos.

### **Pruebas de seguridad**

Las pruebas de seguridad se realizan para comprobar que los mecanismos de protección integrados en el sistema realmente lo protejan de irrupciones inapropiadas (PRESSMAN, 2002). Además, se encargan de certificar que los datos y las funciones del sistema solo son accesibles por los actores debidamente autorizados (Toll, 2007).

Por su parte, la Junta Internacional de Cualificaciones de Pruebas de Software (ISTQB por sus siglas en inglés) define la "Aceptación" aplicado a la rama de la informática como: Pruebas formales que se realizan atendiendo las necesidades del cliente, requerimientos y procesos de negocio, realizadas para determinar si un sistema satisface los requerimientos de aceptación que permitan que el cliente pueda determinar si acepta o no el sistema (Herrerros, 2015).

Para ejecutar estas pruebas, se propone utilizar la herramienta Acunetix Web Vulnerability Scanner.

### **Pruebas de usabilidad**

Según diversos estándares de la Ingeniería de Software, se puede definir la usabilidad como el grado en el que un producto puede ser utilizado por usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción, en un determinado contexto de uso. Como se puede apreciar, la usabilidad de un sistema está ligada a usuarios, necesidades y condiciones específicas (Carcasés, 2016). De manera general, el término usabilidad es empleado para referirse a la capacidad que posee un producto de ser utilizado por los usuarios de forma fácil, eficiente y con satisfacción, en un determinado contexto de uso.

Se puede decir que el proceso de prueba de usabilidad se enfoca en satisfacer las necesidades de los usuarios finales basados en métricas definidas durante la planeación.

Para la realización de las pruebas de usabilidad, se propone el uso de la "Lista de Chequeo de Usabilidad para sitios web", desarrollada por los especialistas del grupo de Seguridad del Departamento de Evaluación de Productos de Software (DEPSW), perteneciente al Centro Nacional de Calidad de Software (CALISOFT).

### **3.4 Conclusiones del capítulo**

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

- I. El diseño del diagrama de secuencia facilitó la visión en cuanto a composición física y lógica del sistema.
- II. Aplicar los estándares de codificación permitió obtener en el sistema un código legible, estándar y fácil de comprender lo que asegura la calidad y facilita un futuro mantenimiento.
- III. El proceso de validación de la solución propuesta a través de las pruebas de carga y estrés, funcionalidad, seguridad, usabilidad y aceptación permitió identificar y corregir las no conformidades detectadas para obtener un producto de mayor calidad.

## **CONCLUSIONES**

1. El estudio de los referentes teóricos y el análisis de las diferentes herramientas y tendencias para la gestión de contenidos permitió determinar la no existencia de un sistema informático que responda a las necesidades requeridas por el cliente.
2. El diseño de la propuesta de solución permitió generar los artefactos más significativos de acuerdo con la metodología de desarrollo de software AUP-UCI tomándose como referencia los requisitos detectados.
3. La implementación del sistema a través de las herramientas y lenguajes seleccionados permitirá obtener una aplicación web capaz de manejar datos referentes a la gestión y el aseguramiento material para la ONBC.
4. Las técnicas de validación definidas para la propuesta de solución permitiera la detección y corrección de las no conformidades detectadas y evidenciaran que el sistema constituye una solución funcional.
5. La validación del problema de investigación mediante la carta de aceptación emitida por el cliente demostrará que el sistema para la gestión y aseguramiento en la ONBC, contribuye a informatizar el proceso de consolidación de los medios materiales en los bufetes colectivos a nivel nacional.

## REFERENCIA BIBLIOGRÁFICA

1. Acunetix. (2021). *Acunetix. Manual de usuario*. <https://www.acunetix.com/wp-content/uploads/2012/10/wvsmanual.pdf>
2. AHORA SGA. (2021). SGA, el Software Avanzado para la Gestión de Almacén. *AHORA*. <https://www.ahora.es/productos/sga/>
3. Alemán de la Torre, L., Padilla Aguiar, D., Cuevas Casas, C. M., Alemán de la Torre, L., Padilla Aguiar, D., & Cuevas Casas, C. M. (2019). Diagnóstico del proceso logístico para la toma de decisiones en empresas de biotecnología. *Retos de la Dirección*, 13(2), 182-202.
4. *Angular*. (2021). <https://angular.io/>
5. Arboleda Aparicio, J. C. (2018). *Libro de investigación Educación, diversidad, lengua y Cultura. Rielec III*. REDIPE. 4- <https://redipe.org/wp-content/uploads/2019/07/Libro-Educacion-diversidad-lengua-y-cultura-rielec-iii-brasil-2018-parte-ii.pdf#page=85>
6. Bruno Sovilla\* y Francisco García Fernández. (2013, abril 24). *La economía cubana: Entre voluntarismo e intentos de planificación (1959-2012)*. <https://www.sciencedirect.com/science/article/pii/S0185191813723079>
7. Budrich GmbH, V. B. (2021). *Políticas sociales y reforma institucional en la Cuba pos-COVID*. <https://library.oapen.org/bitstream/handle/20.500.12657/50323/9783847416951.pdf>
8. Castillo, A. del. (s. f.). *Web dinámicas con PHP*.
9. CEPAL. (2021). *Sistema Nacional de Planificación de Cuba*. <https://observatorioplanificacion.cepal.org/es/sistemas-planificacion/sistema-nacional-de-planificacion-de-cuba>
10. Ceupe. (2021). *Tipos de sistemas logísticos*. Ceupe. <https://www.ceupe.com/blog/tipos-de-sistemas-logisticos.html>
11. DATAZUCAR. (2021). *Start [DATAZUCAR • Empresa de Soluciones Informáticas]*. <https://wikidoku.datazucar.cu/>



12. Exact. (2021). *Exact | Software de gestión para PYMES y Empresas Internacionales*. exact.com. <https://www.exact.com/es>
13. Fortún, M. (2020, junio 3). *Administración financiera*. Economipedia. <https://economipedia.com/definiciones/administracion-financiera.html>
14. Gervacio, L. O. (2018). *Ingeniería y tecnología. Ciencias computacionales y de la información*.
15. GET. (2019). *GET|Productos*. <https://www.get.tur.cu/1>
16. *Get Started with Visual Studio Code*. (s. f.). Recuperado 15 de noviembre de 2021, de <https://code.visualstudio.com/learn/overview>
17. JMeter. (s. f.). *Apache JMeter—Apache JMeter™*. Recuperado 30 de noviembre de 2021, de <https://jmeter.apache.org/>
18. Jose. (2018). *Diseño metodológico para el desarrollo de interfaces gráficas en páginas web utilizando los lenguajes HTML 5 y CSS 3*.
19. Laravel. (2021). *Installation—Laravel—The PHP Framework For Web Artisans*. <https://laravel.com/docs/8.x>
20. Laveriano, W. (2010). *Importancia del control de inventarios en la empresa*. <http://biblioteca.esucomex.cl/RCA/Importancia%20del%20control%20de%20inventarios%20en%20la%20empresa.pdf>
21. León, A. R. (2018). Los sistemas logísticos y las cadenas de suministros. *Anuario Ciencia en la UNAH*, 16(1), Article 1. <https://www.rcta.unah.edu.cu/index.php/ACUNAH/article/view/993>
22. LogísticaMC. (2021). *Sistemas informatizados para la gestión de almacenes*. <https://logisticamc.com/sistemas-informatizados-gestion-almacenes/#>
23. Mecalux. (s. f.). *Flujo logístico: Pieza clave para una cadena de suministro optimizada*. Recuperado 14 de noviembre de 2021, de <https://www.mecalux.com.mx/blog/flujo-logistico>
24. Mecalux. (2021). *Easy WMS*. <https://www.mecalux.com.mx/software/wms-sistema-gestion-almacenes>

25. MOZILLA. (2021). *MDN Web Docs*. <https://developer.mozilla.org/es/docs/Web/>
26. Pérez Sánchez, M. A. (2018, noviembre 19). Versat Sarasola- Un sistema seguro al alcance de la empresa cubana. *Revista TINO*. <https://revista.jovenclub.cu/versat-sarasola-un-sistema-seguro-al-alcance-de-la-empresa-cubana/>
27. *PostgreSQL: About*. (2021). <https://www.postgresql.org/about/>
28. Pressman, R. S. (2010). *Software engineering: A practitioner's approach*. Palgrave macmillan.
29. Rodas. (s. f.). *ALMACÉN – Rodas XXI*. Recuperado 14 de noviembre de 2021, de <http://www.rodasxxi.cu/almacen/>
30. Rodríguez Astudillo. (2018). *Desarrollo de un Sistema Web Multiplataforma de Gestión de Selección de Personal para el área de Talento Humano, utilizando el Framework Bootstrap* [Universidad del Azuay]. <https://dspace.uazuay.edu.ec/handle/datos/7939>
31. Rodríguez, F. (2019, julio 19). *Hacer que otros prosperen*. <http://www.citmatel.inf.cu/noticias/hacer-que-otros-prosperen>
32. Ruano-Ortega, Dr. C. E. R. (2020). *Origen y evolución de la logística en Cuba*. <https://anuarioeco.uo.edu.cu/index.php/aeco/issue/view/318>
33. Sánchez, P. Y. C. (2020). Implementación del patrón arquitectónico MVC en aplicaciones web para la arquitectura del software del sistema de capellanía de la UM. *anuario2020*, 1(1), 112-120.
34. SÁNCHEZ, T. R. (2015). *Metodología de desarrollo para la Actividad productiva de la UCI*.
35. SUITE FACSI+. (2019, abril 29). *SUITE FACSI+*. Empresa de Servicios de Información del Transporte. <https://www.sitrans.cu/productos/suite-facsi>
36. Toll. (2007). *Propuesta de manual de procedimiento de Pruebas de Sistema y su aplicación en el Proyecto CICPC*.

37. Universidad Pedagógica Enrique José, Varona, Cuba, & Fernández Canals, Raúl E.; Carbonell Pérez, Jorge E. (2017). *La relación educación-economía. Una mirada desde las ciencias de la educación.*
38. *Welcome! - The Apache HTTP Server Project.* (s. f.). Recuperado 15 de noviembre de 2021, de <https://httpd.apache.org/>
39. XETID. (2021). *XETID | INFORMÁTICA - AUTOMÁTICA TELECOMUNICACIONES.* <https://www.xetid.cu/producto/38>
40. Yanover, D. A. (2016). <https://www.mastermagazine.com.br/>

## ANEXOS

### A: Lista de Chequeo para requisitos no funcionales

Peso	Indicadores a evaluar	NP	RE	RF
<b>Usabilidad</b>				
crítico	¿Cada campo posee un identificador intuitivo?			
crítico	¿Existe un correcto contraste entre los colores de la aplicación?			
<b>Seguridad</b>				
crítico	¿El sistema cuenta con roles definidos y permisos asociados en correspondencia de estos roles?			
crítico	¿Existe un solo administrador del sistema?			
crítico	¿El acceso al sistema se realiza mediante un proceso de autenticación?			
crítico	¿El sistema valida que la contraseña del usuario tenga un mínimo de 8 caracteres, combinación de letras, números y caracteres especiales?			
crítico	¿Los usuarios tienen acceso a las funcionalidades que corresponden al rol asignado?			
<b>Portabilidad</b>				
	¿El sistema permite acceder y ejecutar las funcionalidades desde el navegador Chrome?			
	¿El sistema permite acceder y ejecutar las funcionalidades desde el navegador Firefox?			
<b>Confiabilidad</b>				
	¿El sistema muestra al usuario mensajes de información como ayuda a las acciones que este debe realizar?			
crítico	¿El sistema muestra mensajes de error explicando los errores que se presentan?			
<b>Mantenibilidad</b>				
crítico	¿El sistema permite la incorporación de nuevas funcionalidades?			

B: Caso de prueba Añadir insumo [Elaboración propia]

<b>Caso de prueba: Añadir insumo</b>
<b>Nombre del requisito:</b> Añadir insumo
<b>Nombre de la persona que realiza la prueba:</b> Maykol Daniel González Matos
<b>Descripción de la prueba:</b> Prueba a la funcionalidad añadir insumo
<b>Entrada/Pasos de ejecución:</b> Se introducen los siguientes datos para crear insumo
<b>Nombre</b> <b>U.M</b> <b>Precio Unitario</b> <b>Sub. Element. Gatos</b> <b>Existencia Inicial Almacén</b> <b>Necesidades</b> <b>Comprar Real</b> <b>Importe</b> <b>Total Distribuir</b>
El administrador presiona el botón Aceptar y si los datos están correctos se crea el insumo que podrá ser visualizado desde la vista Consolidación para los usuarios autenticados en el sistema, si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.

C: Caso de prueba Editar insumo [Elaboración propia]

<b>Caso de prueba: Editar insumo</b>
<b>Nombre de requisito:</b> Editar insumo
<b>Nombre de la persona que realiza la prueba:</b> Maykol Daniel González Matos
<b>Descripción de la prueba:</b> Prueba a la funcionalidad editar insumo
<b>Entrada/Pasos de ejecución:</b> Se modifican los siguientes datos para editar un insumo
<b><u>Datos actuales:</u></b> <b>Nombre:</b> ventiladot <b>U.M:</b> 3 <b>Precio Unitario:</b> 15,5

**Sub. Element. Gatos: 67**

**Existencia Inicial Almacén: 43**

**Necesidades: 2**

**Comprar Real: 47**

**Importe: 6345**

**Total Distribuir: 23**

**Datos después de editados:**

**Nombre: ventilador**

**U.M: 3**

**Precio Unitario: 15,5**

**Sub. Element. Gatos: 67**

**Existencia Inicial Almacén: 43**

**Necesidades: 2**

**Comprar Real: 47**

**Importe: 6345**

**Total Distribuir: 23**

El administrador presiona el botón Aceptar y si los datos están correctos actualiza el insumo que podrá ser visualizado desde la vista Consolidación, si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.